

1-IL CASSETTE & CONTROL FUNCTIONS

ROM ASSEMBLY

REV. 6/81A

INS: L C S

		FILE	SCPL1B	
2				
3	0	34 CON	28	HPIL ID
4	1	52 CON	42	# OF FUNCTIONS
5	2	0 DEFR4K CASSET		
5	3	0		
6	4	0 DEFR4K CREATF		CREAT A FILE
6	5	0		
7	6	0 DEFR4K DIR		CASSETTE DIRECTORY
7	7	0		
8	10	0 DEFR4K NEWTAP		FORMAT A TAPE
8	11	0		
9	12	0 DEFR4K PURGEF		PURGE A FILE
9	13	0		
10	14	0 DEFR4K READA		READ ALL
10	15	0		
11	16	0 DEFR4K READK		READ "KEY" FILE
11	17	0		
12	20	0 DEFR4K READP		READ PROGRAM
12	21	0		
13	22	0 DEFR4K READR		READ ALL REGISTERS
13	23	0		
14	24	0 DEFR4K READRX		READ REGS SEQUENTIALLY BY X
14	25	0		
15	26	0 DEFR4K READS		READ STATUS
15	27	0		
16	30	0 DEFR4K READSB		READ.SUB (APPEND PROG)
16	31	0		
17	32	0 DEFR4K RENAME		RENAME A FILE
17	33	0		
18	34	0 DEFR4K SEC		SECQUIRE A FILE
18	35	0		
19	36	0 DEFR4K SEEKR		SEEK TO REGISTER IN A FILE
19	37	0		
20	40	0 DEFR4K UNSEC		UNSECQUIRE A FILE
20	41	0		
21	42	0 DEFR4K VERIFY		VERIFY A FILE
21	43	0		
22	44	0 DEFR4K WRTA		WRITE ALL
22	45	0		
23	46	0 DEFR4K WRTK		WRITE "KEY" FILE
23	47	0		
24	50	0 DEFR4K WRTP		WRITE PROGRAM
24	51	0		
25	52	0 DEFR4K WRTPV		WRITE PROGRAM PRIVATE
25	53	0		
26	54	0 DEFR4K WTRR		WRITE ALL REGISTERS
26	55	0		
27	56	0 DEFR4K WTRRX		WRITE REGS SEQUENTIALLY BY X
27	57	0		
28	60	0 DEFR4K WRTS		WRITE STATUS
28	61	0		
29	62	0 DEFR4K ZERO		WRITE ZEROS TO A DATA FILE
29	63	0		
30	64	0 DEFR4K COSTNOF		
30	65	0		

31	66	0	DEFR4K	PILPRM	-PIPE FCNS-
31	67	0			
32	70	0	DEFR4K	AUTOIO	AUTO MODE (RESET FLAG 32)
32	71	0			
33	72	0	DEFR4K	FINDID	FIND LAD OF DEVICE BY ID
33	73	0			
34	74	0	DEFR4K	INA	INPUT ALPHA REG
34	75	0			
35	76	0	DEFR4K	INDX	INPUT A ASCII STRING AS A NUMBER
35	77	0			
36	100	0	DEFR4K	INSTAT	INPUT A DEVICE STATUS
36	101	0			
37	102	0	DEFR4K	LISTNR	ADDRESS A DEVICE AS LISTENER
37	103	0			
38	104	0	DEFR4K	LOCAL	GOTO LOCAL
38	105	0			
39	106	0	DEFR4K	MANIO	MANUAL MODE (SET FLAG 32)
39	107	0			
40	110	0	DEFR4K	OUTA	OUTPUT ALPHA REGFLAG 32)
40	111	0			
41	112	0	DEFR4K	PWRDN	GROUP POWER DOWN
41	113	0			
42	114	0	DEFR4K	PWRUP	GROUP POWER UP
42	115	0			
43	116	0	DEFR4K	REMOTE	REMOTE ENABLE
43	117	0			
44	120	0	DEFR4K	SELECT	SELECTED DEVICE
44	121	0			
45	122	0	DEFR4K	STOPIO	INTERFACE CLEAR
45	123	0			
46	124	0	DEFR4K	TRIGER	EXECUTE GROUP TRIGGER
46	125	0			
47	126	0	CON	0	
48	127	0	CON	0	
49	130	223	CON	0223	S
50	131	16	CON	016	N
51	132	6	CON	006	F
52	133	40	CON	040	
53	134	14	CON	014	L
54	135	24	CON	024	T
55	136	3	CON	003	C
56	137	55	CON	055	-
57			PILPRM	ENTRY	PILPRM
58					
59	140	255	CON	0255	
60	141	55	CON	055	
61				ENTRY	CSTNOP
62	142		CSTNOP	1740	RTN
67					

* PILEN - ROUTINE TO ENABLE PIL CHIP

- * 1. WRITE TO R3P/W TO TURN ON OSCILLATOR
- * 2. SET MASTER CLEAR
- * 3. CLEAR MASTER CLEAR
- * 4. ADDRESS SELF AS A SC, CA, LA
- * 5. SEND OUT "ASP 1"

* USED A.X, C +0 SUB LEVEL

* OUTPUT A.X = # OF DEVICES IN THE LOOP

73 ENTRY PILEN

74 ENTRY ASP

7063 76 143 PILEN 116 C=0
 77 144 1110 S9= 1
 78 145 1160 DADD=C
 79 146 1670 C=REGN 14
 80 147 574 RCR 6
 81 150 776 C=C+C S
 82 151 776 C=C+C S
 87 152 1540 RTN C
 84 153 344 C=HPIL 3
 84 154 372
 84 155 303
 85 156 1730 CST EX
 86 157 1204 S7= 0
 87 160 1730 CST EX
 88 161 1300 HPIL=C 3
 89 162 44 HPL=CH 0
 90 163 5 CH= 0001
 91 164 44 HPL=CH 0
 92 165 1611 CH= 0342
 93 166 1104 S9= 0

7063

7063

USER FLAG 33 SET ?
 YES, DON'T DO ANYTHING
 TURN ON CLOCK

SET MASTER CLEAR

SC=CA=TA=1

SEND OUT AN IDY TO CHECK THE LOOP INTEGRITY.
 TIME OUT ONLY SET FOR 1.21 SECONDS.

96 167 144 HPL=CH 1
 97 170 1405 CH= 0301
 98 171 1200 HPIL=C 2
 99 172 460 LDI
 100 173 1750 CON 1000
 101 174 746 C=C+C X
 102 175 IDYTST 1146 C=C-1 X
 103 176 247 GOC AAU25 (222) YES
 104 177 354 ORAV? IDY COMES BACK YET ?
 105 200 1753 GONC IDYTST (175) NOT YET
 106 201 244 C=HPIL 2
 106 202 272
 106 203 203

DO AUTO ADDRESSING ALWAYS

7065 107 204 0 NOP
 108 205 ASP 144 HPL=CH 1
 109 206 1005 CH= 0201
 110 207 244 HPL=CH 2
 111 210 1151 CH= 0232
 112 211 106 C=0 X
 113 212 AAU10 354 ORAV?
 114 213 57 GOC AAU20 (220)
 115 214 0 NOP
 116 215 1046 C=C+1 X
 117 216 47 GOC AAU25 (222)
 118 217 1733 GOTO AAU10 (212)
 119 220 AAU20 1154 FRNS?
 120 221 33 GONC ASP00 (224)
 121 222 AAU25 6 A=0 X
 122 223 633 GOTO SCNDER (306)
 123 224 ASF00 144 HPL=CH 1
 124 225 1205 CH= 0241
 125 226 244 HPL=CH 2
 126 227 1005 CH= 0201
 127 230 106 C=0 X
 128 231 231 ASP10 354 ORAV?
 129 232 57 GOC ASP30 (237) YES

WRITE RDY CONTROL BITS
 SEND ASP

INITIALIZE TIMER

FRAME COMES BACK YET ?

7065

131	233	0 NOP			
132	234	1046 C=C+1	X		TIME OUT YET ?
133	235	1657 GOC	AAU25	(222)	YES
134	236	1733 GOTO	ASP10	(231)	NOT YET
135	237 ASP30	1154 FRNS?			FRAME RETURN SAME AS SEND?
136	240	63 GONC	ASP50	(246)	YES, NOBODY OUT THERE
137	241 ASP40	244 C=HPIL	2		READ THE RETURN FRAME
137	242	272			
137	243	203			
138	244	1146 C=C-1	X		C.X = # OF DEV ON LOOP
139	245	23 GONC	ASP60	(247)	
140	246 ASP50	106 C=0	X		NO DEV ON LOOP
141	247 ASP60	1730 CST EX			
142	250	1204 S7=	0		
143	251	1730 CST EX			
144	252	406 A=C	X		A.X = # OF DEVICES
145	253	1740 RTN			

*
 * SCMD - ROUTINE TO SEND OUT A COMMAND
 *
 * USED C ONLY
 * SUBENTRIES : UNL - SEND UNLISTEN
 * UNT - SEND UNTALK
 * TAD - SEND TALKER ADDRESS (TALKER ADDR IN C.X)
 * LAD - SEND LISTENER ADDRESS (LISTENER ADDR IN C.X)
 * LADA - SAME AS LAD EXCEPT LISTENER IN A.X
 * SNDA - SEND SEC.CMD- "SEND DATA"
 * TALKER - ADDRESS THE DEVICE AS A TALKER (TAD IN R5)
 * LISTENR - ADDRESS THE DEVICE AS A LISTENER (LAD IN R6)
 * SRCHFL - MAKE IT A LISTENER AND SEND "SEARCH"
 * DTFLOW - SEND SEC.CMD - DATA FOLLOW
 *
 * TIME OUT = 4.4 SECONDS FOR ALL FOLLOWING COMMAND FRAMES
 *
 *

164		ENTRY	TALKER
166		ENTRY	SCMD
167		ENTRY	SCMD15
168		ENTRY	SCMD20
169		ENTRY	SCMD30
170		ENTRY	SCMDR
171		ENTRY	SFRMC
172		ENTRY	TAD
173		ENTRY	UNL
174		ENTRY	UNT

*
 704C 176 254 UNT 460 LDI
 177 255 137 CON 0137
 178 256 143 GOTO SCMD (272)
 704F 179 257 UNL 460 LDI
 180 260 77 CON 0077
 181 261 113 GOTO SCMD (272)
 70B2 182 262 TALKER 544 C=HPIL 5
 182 263 572
 182 264 503
 70B5 183 265 TAD 44 HPL=CH 0 SET SC,CA,LA=1
 184 266 1501 CH= 0320
 185 267 1730 CST EX
 186 270 510 S6= 1 TURN ADDR N INTO TAD.N

```

87 271 1730 CST EX
88 272 SCMD 144 HPL=CH 1
89 273 1005 CH= 0201 WRITE CMD CONTROL BITS TO R1W
90 274 SFRMC 1200 HPIL=C 2 WRITE C[0:1] TO R2W
91 275 SCMD15 106 C=0 X INITIALIZE TIMER**CAUTION***
92 276 SCMD20 354 ORAV? FRAME COMES BACK YET ?
93 277 167 GOC SCMD30 ( 315) YES
94 300 0 NOP
95 301 0 NOP
96 302 0 NOP
97 303 0 NOP
98 304 1046 C=C+1 X TIME OUT YET ?
99 305 1713 GONC SCMD20 ( 276) NOT YET
200 306 SCMDER 1110 S9= 1 SET ERROR FLAG
201 307 244 C=HPIL 2 RESET FRNS
201 310 272
201 311 203
202 312 SCMD26 144 HPL=CH 1 TURN OFF FLAG ENABLE
203 313 1 CH= 0000
204 314 1740 RTN
205 315 SCMD30 1154 FRNS? FRAME RETURN NOT AS SEND ?
206 316 1743 GONC SCMD26 ( 312) NO, NO TRANIST ERROR
207 317 1673 GOTO SCMDER ( 306)
208 ENTRY SNDATA
209 ENTRY DTFLOW
210 ENTRY LISTEN
211 ENTRY WRDAT
212 ENTRY SEKSUB
213 ENTRY SRWRT
214 ENTRY DDT0

216 320 DDT0 460 LDI
217 321 300 CON 0300 DDT 0
218 322 1503 GOTO SCMD ( 272)
219 323 SRWRT 460 LDI
220 324 242 CON 0242
221 325 1453 GOTO SCMD ( 272)
222 326 SEKSUB 1 GOSUB SEEKX
223 327 0
223 330 WRDAT 1 GOSUB SRWRT (DDL 2)
223 331 0
224 332 DTFLOW 460 LDI
225 333 240 CON 0240 SAD 00 LISTEN-DATA FOLLOW
226 334 70DD 1363 GOTO SCMD ( 272)
227 335 LISTEN 44 HPL=CH 0
228 336 1601 CH= 0340 SELF AS SC,CA,TA
229 337 544 C=HPIL 5 GET DEV #
229 340 572
229 341 503
230 70E2 ENTRY LAD
231 342 LAD 1730 CST EX
232 343 210 S5= 1 MAKE IT A LISTENER ADDR
233 344 1730 CST EX
234 345 70E6 1253 GOTO SCMD ( 272)
235 346 SNDATA 460 LDI
236 347 300 CON 0300 DDT SAD 00 TALKER-SEND DATA
237 350 1 GOSUB SCMD
237 351 0

```

FALL TO ROUTINE SEND NAT

703A

703E

70D6

70DD

* NATNRD - SEND THE READT FRAME "NAT"
 * AFTER SENDING OUT THE NAT, WAIT FOR THE ORAY TO GET SET.
 * MEANWHILE, CHECK THE KEY BOARD. ANY KEY DOWN WILL CAUSE
 * ABORT THE WAIT LOOP AND RETURN TO MAINFRAME.
 * AS SOON AS ORAY SET, RETURN TO CALLING PROGRAM WITHOUT
 * READING THE INCOMING FRAME.

247 ENTRY NATNRD

70EA

249	352	NATNRD	144	HPL=CH	1	WRITE RDY CONTROLL BITS
250	353		1205	CH=	0241	READY Command
251	354		244	HPL=CH	2	
252	355		601	CH=	0140	CH=SDA SEND NAT
253	356		106	C=0	X	
254	357	NATH10	454	FRAY?		ANY FRAME COMES IN YET ?
255	360		1540	RTN C		YES
256	361		1046	C=C+1	X	TIME OUT YET ?
257	362		547	GOC	RDFMER (436)	YES
258	363		1743	GOTO	NATH10 (357)	NOT YET

* NRD - SEND NOT READY FRAME

1. READ THE LAST FRAME AND SAVE IT IN R6, BUT DON'T ECHO IT.
2. THEN SEND "NRD".
3. AFTER NDR COMING BACK, RETRANSMIT THE LAST DATA FRAME.
4. THEN READ EOT/ETE

* NRDC - SAME AS NRD EXCEPT R2 ALREADY IN C.X

* ASSUME : NOTHING

* OUTPUT : C.X = LAST FRAME

* USED A.X, C, +1 SUB LEVEL

271 ENTRY NRD

272 ENTRY NRDC

70FY

274	364	NRD	1	GOSUB	RDDFRM	READ IN A DATA FRAME
274	365		0			
275	366	NRDC	406	A=C	X	SAVE R2 IN A.X
276	367		144	HPL=CH	1	WRITE RDY CONTROLL BITS
277	370		1205	CH=	0241	
278	371		460	LDI		SEND NRD
279	372		102	CON	0102	
280	373		1	GOSUB	SFRMC	
280	374		0			
281	375		244	C=HPIL	2	GET NRD OUT OF INPUT BUFR
281	376		272			
281	377		203			
282	400		144	HPL=CH	1	WRITE CONTROL BITS FOR DATA FRAME
283	401		5	CH=	0001	
284	402		246	C=A	X	GET LAST DATA FRAME BACK FROM A.X
284	403		406			
285	404		1200	HPIL=C	2	RETRANSMIT LAST DATA FRAME
286	405		1	GOSUB	RDDFRM	READ ETO/ETE
286	406		0			
287	407		1104	S9=	0	
288	410		246	AC EX	X	C.X= LAST DATA FRAME
289	411		1074	ROR	2	A.X= ETO/ETE
290	412		460	LDI		
291	413		100	CON	0100	
292	414		1546	? A#C	X	IS IT AN ETO ?
293	415		217	GOC	RDFMER (436)	NO, MUST BE AN ETE

294 416 1574 RCR 12
295 417 1740 RTN

C.X= LAST DATA FRAME

RDFRM - READ A DATA FRAME

INPUT : ASSUME THE DEVICE IS A LISTENER
IF S9 = 1 WILL RETURN WITHOUT TRYING

OUTPUT : C[1:0] = DATA BYTE

IF TIME OUT WILL SET C=0 & S9=1

IF THE FRAME IS NOT A DATA FRAME WILL SET S9=1

NOTE: THE DATA FRAME WILL NOT BE ECHO BY THIS ROUTINE
USED C +0 SUB LEVEL

307 ENTRY RDDFRM
308 ENTRY RDFMER
309 ENTRY C=R2
310 ENTRY UNLRSF

312 420 RDDFRM 116 C=0 W INITIALIZE TIME OUT COUNTER
313 421 RDFM10 454 FRAV? ANY FRAME COMES IN YET ?
314 422 67 GOC RDFM20 (430) YES
315 423 1046 C=C+1 X CHECK TIME OUT
316 424 1753 GONC RDFM10 (421)
317 425 1076 C=C+1 S
318 426 1733 GONC RDFM10 (421)
319 427 73 GOTO RDFMER (436) TIME OUT
320 430 RDFM20 144 C=HPIL 1 READ CONTROL BITS
320 431 172
320 432 103
321 433 1074 RCR 2
322 434 776 C=C+C S IS THE FRAME A DATA FRAME ?
323 435 23 GONC C=R2 (437) YES
324 436 RDFMER 1110 S9= 1
325 437 C=R2 244 C=HPIL 2 READ DATA BITS
325 440 272
325 441 203
326 442 1740 RTN
327 443 UNLRSF 1 GOSUB UNL
327 444 0
325 445 1723 GOTO C=R2 (437)

SDATA - SEND OUT A DATA FRAME AND SET TIME OUT TO 40 SECONDS

INPUT : C[1:0] = DATA BYTE

OUTPUT : S9 = 1 IF TIME OUT OR "FRNS" SET

USED C +0 SUB LEVEL

335 ENTRY SDATA0
336 ENTRY SDATA

338 446 SDATA0 144 HPL=CH 1
339 447 5 CH= 0001
340 450 SDATA 1200 HPIL=C 2
341 451 116 C=0 W
342 452 SDAT10 354 ORAV?
343 453 67 GOC SDAT20 (461)
344 454 1046 C=C+1 X
345 455 1753 GONC SDAT10 (452)
346 456 1076 C=C+1 S
347 457 1733 GONC SDAT10 (452)
348 460 1563 GOTO RDFMER (436)

```

349 461 SDAT20 1154 FRMS?
350 462          1640 RTN NC
351 463          1533 GOTO   RDFMER ( 436 )

```

```

*
* RDTYPE - READ A DEVICE TYPE
* ASSUME THE DEVICE ALREADY ADDRESSED AS A TALKER, SEND RDY FRAME
* "SAC" AND READ ONE DATA BYTE FROM IT.
* IF THE DEVICE SEND MORE THAN ONE BYTE, ONLY THE FIRST BYTE WILL
* BE KEPT.
* USED A.X, C, S0      NO PT,  +0 SUB LEVEL
* OUTPUT :
*      A[2:1] = STATUS BYTE IF THE DEVICE RESPOND TO SAC
*      A[2:1] = 0 IF THE DEVICE NOT RESPOND TO RDY FRAME - "SAC"

```

```

* DEVICE TYPE HAS BEEN DEFINED AS FOLLOW :

```

```

* FILBERT   - HEX 10
* SPECIAL.K - HEX 20
* WALLABY   - HEX 30
* PILBOX    - HEX 40
* PLOTTER    - HEX 50

```

```

369          ENTRY  RDTYPE
370          ENTRY  RDTYPC
371          ENTRY  RTPHRT

```

```

*
373 464 RDTYPE 460 LDI
374 465          143 CON      @143          SAC
375 466 RDTYPC 144 HPL=CH 1          WRITE RDY CONTROL BITS
376 467          1205 CH=      @241
377 470          1200 HPIL=C 2
378 471          6 A=0      X
379 472          1610 S0=      1
380 473 RTYP10 106 C=0      X          SET TIMER
381 474 RTYP20 354 ORAY?          FRAME RETURN YET ?
382 475          57 GOC      RTYP30 ( 502 ) YES
383 476          0 NOP
384 477          1046 C=C+1    X          TIME OUT ?
385 500          1367 GOC      RDFMER ( 436 ) YES
386 501          1733 GOTO     RTYP20 ( 474 ) NOT YET
387 502 RTYP30 144 C=HPIL 1          SEE WHAT FRAME IS COMMING BACK?
388 503          172
389 504          103
390 505          1374 RCR      13
391 506          746 C=C+C      X
392 507          1307 GOC      C=R2 ( 437 ) NOT DATA FRAME, READ R2
393 510          1 GOLONG RTPACH      READ THE RETURN DATA BYTE
394 511          2
395 512 RTPHRT 1614 ?S0=1          FIRST BYTE ?
396 513          1603 GONC      RTYP10 ( 473 ) NO, IGNORE IT
397 514          406 A=C      X
398 515          1746 A SL      X
399 516          1604 S0=      0
400 517          1543 GOTO     RTYP10 ( 473 ) READ NEXT FRAME

```

```

* GETDEV - GET SELECTED OR DEFAULT DEVICE #

```

```

* LOGIC:

```

```

* 1. FRIENDLY LOOP - UNFRIENDLY FLAG(USER FLAG 32) NOT SET
*   USE SELECTED DEVICE # OR DEFAULT TO 1 IF NO DEVICE BEEN SELECTED
* 2. UNFRIENDLY LOOP
*   USE SELECTED DEVICE #, DEFAULT IS AN ERROR.
* INPUT : A.X = # OF DEVICES IN LOOP

```


OUTPUT : C.X = START SEARCHING DEVICE #
 R5R/W = START SEARCHING DEVICE #
 R6R/W = R5R/W IF IN FRIENDLY MODE
 = R5R/W +1 IF IN UNFRIENDLY MODE
 USED C, S0-7, +0 SUB LEVEL

		ENTRY	GETDEV	
412				
414	520	106	C=0 X	
415	521	1160	DADD=C	ENABLE CHIP 0
416	522	1670	C=REGN 14	LOAD USER FLAG 32
417	523	274	ROR 5	
418	524	1530	ST=C	S3 = FLAG 32
419	525	444	C=HPIL 4	GET SELECTED DEVICE #
419	526	472		
419	527	403		
420	530	1406	? ACC X	STRAT # > # OF DEVICE ?
421	531	33	GONC GTD#20 (534)	NO, END = START # +1
422	532	106	C=0 X	YES, END # = START # = 1
423	533	1046	C=C+1 X	
424	534	1500	HPIL=C 5	
425	535	1600	HPIL=C 6	
426	536	1740	RTN	
427			FILLTO 0540	
	537	0000	NOP	
	540	0000	NOP	

NXTDEV - ROUTINE TO GET NEXT DEVICE #

LOGIC :

1. IF IN UNFRIENDLY MODE, RETURN TO P+1 IMMEDIATELY, DEVICE NOT FOUND.
2. IF CURRENT SEARCHING DEVICE # < # OF DEVICES IN LOOP, NEXT DEVICE # = CURRENT DEVICE + 1
3. IF CURRENT SEARCHING DEVICE # >= # OF DEVICES IN LOOP, NEXT DEVICE # = 1
4. IF NEXT DEVICE # = START SEARCHING DEVICE #, DEVICE NOT FOUND, RETURN TO P+1
5. IF NEXT DEVICE # NOT EQUAL STARTING DEVICE #, RETURN TO P+2 WITH NEXT DEVICE # IN C.X

INPUT : A.X = # OF DEVICES IN THE LOOP
 R5R/W = THE DEVICE # CURRENTLY BEING POLL
 R6R/W = START SEARCHING DEVICE #

OUTPUT: C.X = NEXT SEARCHING DEVICE #
 USED A.X, C, S0-7, +0 SUB LEVEL

		ENTRY	NXTDEV	
449				
449			NXTDEI	
451	541	1670	C=REGN 14	
452	542	574	ROR 6	
453	543	776	C=C+C 8	IN UNFRIENDLY MODE ?
454	544	1540	RTN C	YES, DON'T GO TO NEXT DEVICE
455	545	544	C=HPIL 5	GET CURRENT DEV #
455	546	572		
455	547	503		
456	550	1046	C=C+1 X	POINT TO NEXT DEV #
457	551	1406	? ACC X	WRAP AROUND THE LOOP YET ?
458	552	33	GONC NXTD30 (555)	NOT YET
459	553	106	C=0 X	YES, START FROM DEV #1 AGAIN

```

460 554          1046 C=C+1  X
461 555 NXTD30   406 A=C    X
462 556          644 C=HPIL 6
462 557          672
462 560          603
462 561          246 AC EX   X
464 562          1500 HPIL=C 5
465 563          1546 ? A#C  X
466 564          1640 RTN NC
467 565          713 GOTO   RTNP+2 ( 656)

```

GET STARTING DEV #

SEARCH ENTIRE LOOP YET ?
YES, DEVICE NOT FOUND

```

*
* FNDPTR - FIND A PRINTER IN A PIL LOOP
* FNDCAS - FIND A CASSETTE IN A LOOP
*
* ASSUME : CHIP 0 ENABLE
* USED A, C, S0-7, NO PT, +1 SUB LEVEL
* OUTPUT :
* IF PRINTER OR CASSETTE FOUND :
*   1. RETURN TO P+2
*   2. SAVE DEVICE ADDR IN R5R/W
*   3. S9 = 0
* IF PRINTER OR CASSETTE NOT FOUND :
*   1. RETURN TP P+1
*   2. S9 = 1
*

```

```

483          ENTRY  FNDPTR
484          ENTRY  FNDCAS
485          ENTRY  RTNP+2
486          ENTRY  FDEV20

```

```

*
7176 488 566 FNDCAS      1 GOSUB  PILEN 7063
488 567          0
489 570          36 A=0      S
490 571          576 A=A+1    S
491          LEGAL
492 572          1 GOSUB  PLERCK 7067 CHECK LOOP INTACK
492 573          0
493 574          53 GOTO   FNDDEV ( 601)
717D 494 575 FNDPTR      36 A=0      S
495 576          1140 SETHEX
496 577          1 GOSUB  PILEN      ENABLE PIL CHIP
496 600          0
7181 497 601 FNDDEV      1114 ?S9=1    FRAME GO THRU THE LOOP ?
498 602          1540 RTN C      NO
499 603          674 RCR      11
500 604          432 A=C      M      SAVE # OF DEVS IN A.M
501 605          1 GOSUB  GETDEV 7065 GET START SEARCHING DEVICE #
501 606          0
502 607 FDEV10      1 GOSUB  TAD      7065 ADDRESS THE DEVICE AS A TALKER
502 610          0
503 611          1 GOSUB  RDTYPE 7067 READ THE DEVICE TYPE
503 612          0
504 613          460 LDI
505 614          400 CON      0400 C.X = HEX 100
506 615          1536 ? A#0    S      LOOKING FOR PRINTER ?
507 616          133 GONC     FRTR35 ( 631) YES
508 617          1546 ? A#C    X      IS A FILBER TYPE MASS STORAGE ?
509 620          323 GONC     FDEV40 ( 652) YES, SIMILIAR TO FILBER
510 621 FDEV20      256 C=A      W      GET # OF DEVICES IN LOOP
510 622          416

```

```

511 623          74 RCR      3
512 624          406 A=C     X
513 625          1 GOSUB    NXTDEV 716A GET NEXT DEVICE # & CHECK DONE
513 626          0
514 627          323 GOTO     FDEV50 ( 661 ) DEVICE NOT FOUND
515 630          1573 GOTO    FDEV10 ( 607 ) KEEP SEARCHING
516 631 FRTR35   746 C=C+C   X      C.X = HEX 200
517 632          1546 ? A#C   X      IS IT THE SPECIAL-K ?
518 633-         113 GONC     FDEV38 ( 644 ) YES
519 634          1566 ? A#C   XS     OTHER KIND OF PRINTER ?
520 635          43 GONC      FDEV36 ( 641 ) YES
521 636          1066 C=C+1   XS     C.XS = 3
522 637          1566 ? A#C   XS     SOME DISPLAY ?
523 640          1617 GOC      FDEV20 ( 621 ) NO
524 641 FDEV36   106 C=0      X      YES, DON'T ASK ITS STATUS
525 642          1146 C=C-1   X      ALL 1'S IN R6 SAY T.V.
526 643          1600 HPIL=C   6
527 644 FDEV38   1 GOSUB     LDSST0 SEE IF PTR EXISTING FLAG SET?
527 645          0
528 646          1614 ?S0=1
529 647          37 GOC       FDEV40 ( 652 ) YES, IT IS O.K.
530 650          1 GOSUB     SF51 SET FLAG 55&21
530 651          0
531 652 FDEV40   1 GOSUB     FNSTSA 71C3 READ THE STATUS 71AA
531 653          0
532 654          1114 ?S9=1
533 655          77 GOC       FDEV60 ( 664 )
534 656 RTNP+2   660 C=STK
535 657          1072 C=C+1   M.
536 660          740 GOTOC
537 661 FDEV50   1670 C=REGN  14     IN MANUAL MODE ?
538 662          574 RCR      6
539 663          776 C=C+C    S
540 664 FDEV60   1 GOLNC     UNT +0AC NOT IN MANUAL MODE
540 665          2
541 666          1536 ? A#0    S     LOOKING FOR PRINTER ?
542 667          1523 GONC     FDEV36 ( 641 ) YES
543 670          1623 GOTO     FDEV40 ( 652 )

```

PATCH FOR "READ TYPE"(RDTYPE) ROUTINE

```

547          ENTRY RTPACH
548 671 RTPACH   244 C=HPIL  2     READ THE STATUS BYTE FROM R2
548 672          272
548 673          203
549 674          1200 HPIL=C  2     ECHO IT
550 675          1 GOLONG RTPHRT RETURN TO "RDTYPE" ROUTINE
550 676          2
551          FILLTO 0677
551 677          0000 NOP

```

```

*
* FNSTS - FETCH NEW DEVICE STATUS
* FNSTS1 - SAME AS FNSTS EXCEPT ONLY READ ONE BYTE
*
* ASSUME : R5R/W = DEVICE LOOP ADDRESS
* INPUT :
* IF S0 = 1 READ TWO BYTES OF STATUS
* IF S0 = 0 READ ONE BYTE OF STATUS
* USED A, C, NO PT, +0 SUB LEVEL
* OUTPUT :

```

```

* A. FNSTS -
* 1. ORIGINAL ST[7:0] IN CI[1:0]
* 2. 1ST BYTE OF STATUS IN S[7:0]
* 3. 2ND BYTE OF STATUS IN CI[13:12]
* 4. BIT 3 OF 2ND STATUS BYTE IS SAVED IN BIT 3 OF R3R/W
* 5. ADDRESS THE DEVICE AS A LISTENER
* 6. SELF AS A TALKER
* B. FNSTS1 -
* 1. ORIGINAL S[7:0] IN CI[1:0]
* 2. STATUS BYTE IN S 0-7
* 3. ADDRESS THE DEVICE AS A LISTENER
* 4. SELF AS A TALKER

```

```

575          ENTRY  FNSTS
576          ENTRY  FNSTSA
577          ENTRY  CSSTAS

```

```

579 700 CSSTAS 1604 S0= 0
580 701          23 GOTO FNSTSA ( 703 )
581 702 FNSTS 1610 S0= 1
582 703 FNSTSA 16 A=0 W
583 704          1630 C=ST
584 705          1114 ?S9=1
585 706          1540 RTN C
586 707          44 HPL=CH 0
587 710          1501 CH= 0320
588 711          144 HPL=CH 1
589 712          1005 CH= 0201

```

```

ERROR SO FAR ?
YES, DON'T EVEN TRY
SAY I'M SC,CA,LA
GET READY TO SEND CMD

```

```

* TEST IF TALKING TO A DUMM T.V.

```

```

591 713          644 C=HPIL 6
591 714          672
591 715          603
592 716          1166 C=C-1 XS
593 717          1046 C=C+1 X
594 720          263 GONC FSP10 ( 746 )
595 721          1160 DADD=C
596 722          460 LDI
597 723          17 CON 0017
598 724          416 A=C W
599 725          1670 C=REGN 14
600 726          1174 RCR 9
601 727          1730 CST EX
602 730          114 ?S4=1
603 731          23 GONC FSP04 ( 733 )
604 732          566 A=A+1 XS
605 733 FSP04 14 ?S3=1
606 734          33 GONC FSP06 ( 737 )
607 735          566 A=A+1 XS
608 736          566 A=A+1 XS
609 737 FSP06 1730 CST EX
610 740          256 AC EX W
611 741          1474 RCR 1
612 742          1700 HPIL=C 7
613 743          1074 RCR 2
614 744          416 A=C W
615 745          323 GOTO FSPEX4 ( 777 )
616 746 FSP10 544 C=HPIL 5
616 747          572
616 750          503
617 751          1730 CST EX

```

```

TALKING TO A T.V. ?
NO
IF USER FLG15 - TRACE MODE
IF USER FLG16 SET - NORMAL MODE
TO TRACE MODE
TRACE MODE ?(FLAG 15 SET)
NO
A.XS = 1
NORMAL MODE ?(FLAG 16 SET)
NO
A.XS = 2 OR 3
GET DEVICE LOOP ADDRESS

```

618	752	510	S6=	1	MAKE IT A TALKER ADDR
619	753	1730	CST EX		
620	754	1200	HPIL=C	2	& MAKE HIM A TALKER
621	755	FSP30	354	ORAV?	READY TO SEND NEXT CMD ?
622	756	47	GOC	FSP45 (762)	YES
623	757	1046	C=C+1	X	TIME OUT ?
624	760	137	GOC	FSPER (773)	YES
625	761	1743	GOTO	FSP30 (755)	NOT YET
626	762	FSP45	144	HPL=CH	1
627	763	1205	CH=	0241	SEND SSP
628	764	244	HPL=CH	2	
629	765	605	CH=	0141	
630	766	FSP48	106	C=0	X
631	767	FSP50	454	FRAV?	READY TO SEND NEXT CMD ?
632	770	277	GOC	FSP60 (1017)	YES
633	771	1046	C=C+1	X	TIME OUT ?
634	772	1753	GONC	FSP50 (767)	NOT YET
635	773	FSPER	1110	S9=	1
636	774	1073	GOTO	FNSTSA (703)	
637	775	FSPEX	1614	?S0=1	FOR PRINTER ?
638	776	553	GONC	FSP85 (1053)	NO
639	777	FSPEX4	44	HPL=CH	0
640	1000	1611	CH=	0342	SAY I'M A SC,CA,TA
641	1001	144	HPL=CH	1	
642	1002	1005	CH=	0201	WRITE CMD CONTROL BITS
643	1003	544	C=HPIL	5	GET DEVICE LOOP ADDR
643	1004	572			
643	1005	503			
644	1006	1730	CST EX		
645	1007	210	S5=	1	MAKE IT LISTENER ADDR
646	1010	1730	CST EX		
647	1011	1200	HPIL=C	2	ADDRESS THE DEVICE AS A LISTENER
648	1012	FSPEX6	354	ORAV?	READY FOR NEXT FRAME ?
649	1013	367	GOC	FSP80 (1051)	YES
650	1014	1046	C=C+1	X	TIME OUT ?
651	1015	1567	GOC	FSPER (773)	YES
652	1016	1743	GOTO	FSPEX6 (1012)	NOT YET
653	1017	FSP60	244	C=HPIL	2
653	1020	272			READ ONE BYTE
653	1021	203			
654	1022	1200	HPIL=C	2	ECHO
655	1023	1614	?S0=1		PRINTER ?
656	1024	123	GONC	FSP68 (1036)	NO
657	1025	1536	? A#0	S	HAS TO READ ANOTHER BYTE ?
658	1026	57	GOC	FSP65 (1033)	NO
659	1027	576	A=A+1	S	REMEMBER ALREADY READ 1 BYTE
660	1030	406	A=C	X	SAVE 1ST BYTE IN A.X
661	1031	1700	HPIL=C	7	SAVE 1 BYTE OF STS IN R7
662	1032	1343	GOTO	FSP48 (766)	GOTO READ NEXT BYTE
663	1033	FSP65	1600	HPIL=C	6
664	1034	1074	RCR	2	SAVE BIT 3 OF 2ND BYTE TO R6
665	1035	246	AC EX	X	CD[13:0] = 1ST BYTE
666	1036	FSP68	1074	RCR	2
667	1037	416	A=C	W	CD[13:10] = 1ST & 2ND BYTE
668	1040	FSP70	454	FRAV?	SAVE WHOLE THING IN A TEMP.
669	1041	47	GOC	FSP75 (1045)	ETO OR ETE COME IN YET ?
670	1042	1046	C=C+1	X	YES
671	1043	1307	GOC	FSPER (773)	TIME OUT YET ?
672	1044	1743	GOTO	FSP70 (1040)	YES
673	1045	FSP75	244	C=HPIL	2
					NOT TIME OUT YET
					READ ETO/ETE

```

673 1046          272
673 1047          203
674 1050          1253 GOTO   FSPEX  ( 775)
675 1051 FSP80     1154 FRNS?          FRAME RETURN NOT AS SEND ?
676 1052          1217 GOC    FSPER  ( 773) YES, ERROR
677 1053 FSP85     256 AC EX   W
678 1054          53 GOTO    FS100  (1061)

```

```

* OOPCHK (OUT-OF-PAPER CHECK) - CHECKS FOR OUT-OF-PAPER STATUS
* AND PUTS SECOND STATUS BYTE INTO ST[7:0]
*
* ON ENTRY ASSUMES FIRST BYTE OF PRINTER STATUS IS IN ST[7:0]
* IF OOPS THEN SETS S9.
* ALWAYS PUTS WHAT WAS IN C[13:12] ON ENTRY INTO ST[7:0]
* AND PUTS FIRST PRINTER STATUS BYTE TO C[3:2]. C[1:0] IS
* PRESERVED.

```

```

691          ENTRY  FS100
692          ENTRY  OOPCHK
693 1055 OOPCHK     14 253=1          OOPS?
694 1056          23 GONC    OOP10  (1060) NO
695 1057          1110 S9=   1        YES, SET ERROR FLAG.
696 1060 OOP10     1730 CST EX
697 1061 FS100     1574 RCR    12
698 1062          1730 CST EX
699 1063          1740 RTN

```

```

*****
* OUTA - SEND ALPHA REG OUT THROUGH PIL
* AOUTFL - SAME AS AOUT1 EXCEPT WILL PICK UP THE NAME FROM THE
* ADDR SAVED IN REG.8 [13:10]
* AOUTIH - PUT THE BEGINNING ADDR OF ALPHA REGISTER IN REG.8[13
* :10]
* THE FILE NAME WILL BE TERMINATED BY EITHER A COMMA OR END OF
* ALPHA REGISTER. A FILE NAME HAS TO BE EXACTLY 7 CHARS. IT WILL
* BE TRUNCATED OR FILLED WITH TRAILING BLANKS IF IS LONGER OR
* SHORTER THAN 7 CHARS. THE ADDR OF A DELIMINATOR WILL BE SAVED
* IN REG.8[13:10] EVERY TIME FOR LATER USE
*
* USED A,B[3:0],C,S7,S6,S5,S4 +1 SUB LEVEL
* STATUS USED :
* S7 : IF S7=0 SEND OUT ENTIRE ALPHA REG(FOR AOUT FUNCTION)
* IF S7=1 GET FILE NAME FROM ALPHA REG
* S6 : SET TO "0" AT BEGINNING, WILL BE SET TO "1" FOR ANY
* NON-NULL CHAR IN ALPHA REGISTER
* S5 : SET TO "0" AT BEGINNING, WILL BE SET TO "1" IF A COMMA
* IS ENCOUNTERED
* S3 :
* S3=1 THE FILE WILL BE SHIFTED INTO M FROM LEFT END
* S3=0 // RIGHT END
* S2 : IF S2=1, LAST CHAR ADDR WILL BE SAVED IN REG.8[13:10]
* S2=0, DON'T CHANGE REG.8 AT ALL
*****

```

```

728          ENTRY  OUTA
729          ENTRY  AOUT1
730          ENTRY  AOUTIH
731          ENTRY  AOUTFL

```

7234

733	1064	AOUT1	1	GOSUB	AOUTIN	
733	1065		0			
734	1066	AOUTFL	1210	S7=	1	SET FLAG TO GET FILE NAME — 7236
735	1067		504	S6=	0	CLEAR NULL STRING FLAG
736	1070		204	S5=	0	RESET COMMA FLAG
737	1071		1070	C=REGN	8	
738	1072		34	PT=	3	
739	1073		374	ROR	10	
740	1074		352	BC EX	WPT	PUT LAST ADDR IN "B"
741	1075		1334	PT=	13	
742	1076		720	LC	7	
743	1077		276	AC EX	5	CHAR COUNTER IN A(13)
744	1100		34	PT=	3	
745	1101		633	GOTO	AOUT20 (1164)	
747	1102		201	CON	0201	A
748	1103		24	CON	024	T
749	1104		25	CON	025	U
750	1105		17	CON	017	O
751	1106	OUTA	1	GOSUB	SCHDEV	72DD SEARCH THE G.P. INTERFACE MODULE
751	1107		0			
752	1110		1	GOSUB	LISTEN	70DD
752	1111		0			
753	1112		1704	CLR ST		
754	1113		1	GOSUB	AOUTIN	SET STARTING ADDR
754	1114		0			
755	1115	AOUT10	1	GOSUB	INCADA	GET NEXT BYTE
755	1116		0			
756	1117		1	GOSUB	GTBYTA	
756	1120		0			
757	1121		212	B=A	WPT	ADDR TO B
758	1122		1434	PT=	1	
759	1123		1352	? C#0	WPT	IS IT A LEADING BLANK?
760	1124		403	GONC	AOUT20 (1164)	YES
761	1125		510	S6=	1	SET NON-NULL STRING FLAG
762	1126		1214	?S7=1		OUTPUTING FILE NAME ?
763	1127		333	GONC	AOUT18 (1162)	NO, SEND IT
764	1130		412	A=C	WPT	CHECK FOR COMMA
765	1131		460	LDI		
766	1132		54	CON	054	COMMA
767	1133		1552	? A#C	WPT	IS CHAR A COMMA?
768	1134		37	GOC	AOUT12 (1137)	NO
769	1135		210	S5=	1	
770	1136		663	GOTO	AOUT32 (1224)	
771	1137	AOUT12	1536	? A#0	S	7 CHARS YET ?
772	1140		243	GONC	AOUT20 (1164)	YES
773	1141		252	AC EX	WPT	
774	1142	AOUT14	676	A=A-1	S	DEC. CHAR COUNT
775	1143	AOUT15	1434	PT=	1	
776	1144		412	A=C	WPT	
777	1145		630	C=M		
778	1146		14	?S3=1		SHIFT TO M FROM LEFT ?
779	1147		43	GONC	AOUT16 (1153)	NO, FROM RIGHT
780	1150		252	AC EX	WPT	SHIFT CHAR TO M FROM LEFT
781	1151		1074	ROR	2	
782	1152		63	GOTO	AOUT17 (1160)	
783	1153	AOUT16	256	AC EX	W	SHIFT CHAR TO M FROM RIGHT
784	1154		1756	A SL		
785	1155		1756	A SL		

786	1156	412	A=C	WPT	
787	1157	256	AC EX	W	
788	1160	530	M=C		
789	1161	33	GOTO	ROUT20 (1164)	
790	1162	1	GOSUB	SDATA0	NO, SEND CHAR
790	1163	0			
791	1164	34	PT=	3	
792	1165	152	AB EX	WPT	ADDR BACK TO A.
793	1166	460	LDI		
794	1167	5	CON2	0	5
795	1170	102	C=0	PT	
796	1171	1552	? A#0	WPT	END OF "A" REG.
797	1172	1237	GOC	ROUT10 (1115)	NOT YET
798	1173	212	B=A	WPT	ADDR TO B
799	1174	514	?S6=1		NULL STRING ?
800	1175	67	GOC	ROUT30 (1203)	NO
801	1176	1214	?S7=1		OUTPUTING FILE NAME ?
802	1177	233	GONC	ROUT31 (1222)	NO
803	1200	14	?S3=1		SHIFT FROM LEFT ?
804	1201	553	GONC	FLNMER (1256)	NO, SAY ERR
805	1202	1740	RTN		
806	1203	1214	?S7=1		OUTPUTING FILE NAME ?
807	1204	207	GOC	ROUT32 (1224)	YES
808	1205	1670	C=REGN	14	CHECK USER FLAG 17
809	1206	374	ROR	10	IF IT IS SET, SUPPRESS CR,LF
810	1207	776	C=C+C	S	
811	1210	776	C=C+C	S	
812	1211	117	GOC	ROUT31 (1222)	
813	1212	460	LDI		
814	1213	15	CON	13	
815	1214	1	GOSUB	SDATA	SEND "CR"
815	1215	0			
816	1216	460	LDI		
817	1217	12	CON	10	
818	1220	1	GOSUB	SDATA	SEND "LF"
818	1221	0			
819	1222	1	GOLONG	UNLCHK	7336
819	1223	2			
820	1224	1536	? A#0	S	7 CHARS YET ?
821	1225	153	GONC	ROUT40 (1242)	YES
822	1226	630	C=M		
823	1227	14	?S3=1		SHIFT FROM LEFT ?
824	1230	53	GONC	ROUT35 (1235)	NO, FROM RIGHT
825	1231	1074	ROR	2	
826	1232	530	N=C		
827	1233	676	A=A-1	S	
828			LEGAL		
829	1234	1703	GOTO	ROUT32 (1224)	
830	1235	1574	ROR	12	SHIFT BLANK TO M FROM RIGHT
831	1236	1434	PT=	1	
832	1237	220	LC	2	
833	1240	20	LC	0	
834	1241	1713	GOTO	ROUT33 (1232)	
835	1242	1014	?S2=1		NEED TO SAVE ADDR ?
836	1243	113	GONC	ROUT55 (1254)	NO
837	1244	34	PT=	3	
838	1245	152	AB EX	WPT	
839	1246	1070	C=REGN	8	
840	1247	374	ROR	10	
841	1250	252	C=W	WPT	

841	1251		412		
842	1252		174	ROR	4
843	1253		1050	REGN=C	8
844	1254	AOUT55	1	GOLONG	FLERCK
844	1255		2		

STO ADDR IN R(10-13)
ERROR CHECK AND RETURN

846				ENTRY	FLNMER	
847	1256	FLNMER	1	GOSUB	FLEREX	72AC
847	1257		0			
848	1260		16	CON	016	N
849	1261		1	CON	001	A
850	1262		15	CON	015	M
851	1263		1005	CON	01005	E
852	1264	DSPERJ	1	GOLONG	DSPERR	
852	1265		2			

854	1266	AOUTIN	116	C=0		
855	1267		34	PT=	3	
856	1270		620	LC	6	
857	1271		1634	PT=	0	
858	1272		1020	LC	8	
859	1273		34	PT=	3	
860	1274		412	A=C	WPT	
861	1275		1104	S9=	0	
862	1276		1503	GOTO	AOUT50 (1246)	

* SELD - SELECT DEVICE # *

* GET THE INTEGER PART IN X-REG AS THE DEVICE #, CONVERT IT TO *

* BINARY AND STORE IT IN R7R/W. *

* THE NUMER HAS TO BE BETWEEN 1 AND 30 *

873			ENTRY	SELECT
874			ENTRY	CHKLAD
875			ENTRY	LADERM

877	1277		224	CON	0224	T
878	1300		3	CON	003	C
879	1301		5	CON	005	E
880	1302		14	CON	014	L
881	1303		5	CON	005	E
882	1304		23	CON	023	S
883	1305	SELECT	1	GOSUB	CONV30	CONVERT X TO BINARY
883	1306		0			
884	1307		410	S8=	1	
885	1310	CHKLAD	1346	? C#0	X	LAD = 0 ?
886	1311		113	GONC	LADER (1322)	YES
887	1312		406	A=C	X	
888	1313		460	LDI		
889	1314		37	CON	31	
890	1315		1406	? A<C	X	LAD < 31 ?
891	1316		43	GONC	LADER (1322)	NO
892	1317		246	AC EX	X	
893	1320	SELD10	1400	HPIL=C	4	
894	1321		1740	RTH		
895	1322	LADER	414	?S8=1		DISPLAY ERR MESSAGE ?
896	1323		47	GOC	LADERM (1327)	YES

7208

```

897 1324          106 C=0      X
898 1325          1046 C=C+1    X          SET SELECTED DEV # TO 1
899              LEGAL
900 1326          1723 GOTO     SELD10 (1320)
901 1327 LADERM    1 GOSUB     FLEREX
901 1330          0
902 1331          1 CON        001          A
903 1332          4 CON        004          D
904 1333          1022 CON      01022       R
905 1334          1303 GOTO     USPERJ (1264)

```

```

*
*
* SCHDEV - GET SELECT DEVICE NUMBER

```

```

* USED: A,X, C, +1 SUB LEVEL
* OUTPUT : R5 = DEVICE #

```

```

*
* 913              ENTRY  SCHDEV
*
* 915 1335 SCHDEV    1 GOSUB  FILEN
* 915 1336          0
* 916 1337          1 GOSUB  FLERCK      CHECK IF ANY ERROR
* 916 1340          0
* 917 1341          1 GOLONG GETDEV      GET START SEARCHING DEV #
* 917 1342          2

```

```

*
*****

```

```

*+
* SF5521 - SET FLAGS 55 AND 21
* USES: C, ST, AND 1 ADDITIONAL SUBROUTINE LEVEL
* IN: NOTHING
* OUT: SS 0 UP, CHIP 0 ENABLED, C=REG 14
* ASSUMES: NOTHING

```

```

*
* 927              ENTRY  SF5521
* 928              ENTRY  SF51
* 929 1343 SF5521    1 GOSUB  LDSST0      GET FLAGS
* 929 1344          0
* 930 1345 SF51      1610 S0=      1      SET PRINTER EXISTANCE FL
* 931 1346          1630 C=ST
* 932 1347          474 RCR        8
* 933 1350          1730 CST EX      C= SST 0
* 934 1351          1010 S2=      1      SET PRINTER CONTROL FL/
* 935 1352          1730 CST EX      (ENABLE PRINTER)
* 936 1353          574 RCR        6
* 937 1354 SFRTN    1650 REGN=C 14      PUT FLAGS BACK
* 938 1355          763 GOTO     SETIDY (1413)

```

```

*
*****

```

```

* MANIO -- SET UNFRIENDLY FLAG - USER FLAG 32
* AUTOIO - RESET UNFRIENDLY FLAG - USER FLAG 32

```

```

*
*****

```

```

*
* 946              ENTRY  MANIO
* 947              ENTRY  AUTOIO
* 948              ENTRY  RSIDY

```

```

*
* 950 1356          217 CON        0217      0
* 951 1357          11 CON        011       1

```

952	1360	16	CON	@16	N
953	1361	1	CON	@01	A
954	1362	15	CON	@15	M
955	1363	MANIO	410	S8=	1
956	1364		103	GOTO	FLAG32 (1374)
957	1365		217	CON	@217
958	1366		11	CON	@11
959	1367		17	CON	@17
960	1370		24	CON	@24
961	1371		25	CON	@25
962	1372		1	CON	@01
963	1373	AUTOIO	404	S8=	0
964	1374	FLAG32	1670	C=REGN	14
965	1375		274	ROR	5
966	1376		1730	CST EX	
967	1377		4	S3=	0
968	1400		414	?S8=1	
969	1401		23	GONC	*+2 (1403)
970	1402		10	S3=	1
971	1403		1730	CST EX	
972	1404		1174	ROR	9
973	1405		1650	REGN=C	14
974	1406		414	?S8=1	
975	1407		1540	RTN C	
976	1410	RSIDY	1	GOSUB	FNDPTR
976	1411		0		
977	1412		1740	RTN	
978	1413	SETIDY	344	HPL=CH	3
979	1414		401	CH=	@100
980	1415		1	GOLONG	UHL
980	1416		2		

GO INTO AUTO MODE ?
NO
SEE IF PRINTER PLUG-IN
NO
SET AUTO IDY

 REMOTE - SEND COMMAND "REMOTE ENABLE"
 TRIGER - SEND COMMAND "GROUP EXECUTE TRIGGER"
 NOTES :
 1. ALL ABOVE FUNCTIONS WILL SEARCH FOR G.P.I.O. IF IN AUTO MODE, OTHERWISE WILL USE SELECTED DEVICE
 2. ALL ABOVE FUNCTIONS WILL ADDRESS THE SELECTED DEVICE AS A LISTENER BEFORE SENDING THE COMMAND AND THEN SEND AN UNLISTEN COMMAND AFTERWARD.

994	ENTRY	REMOTE
995	ENTRY	LOCAL
996	ENTRY	TRIGER
997	ENTRY	UNLCHK

1000	1417	322	CON	@222	R
1001	1420	5	CON	@05	E
1002	1421	7	CON	@07	G
1003	1422	7	CON	@07	G
1004	1423	11	CON	@11	I
1005	1424	22	CON	@22	R
1006	1425	24	CON	@24	T
1007	1426	TRIGER	460	LDI	
1008	1427		10	CON	@010
1009	1430		263	GOTO	SNOMD (1456)

EXECUTE GROUP TRIGGER

1011	1431	214	CON	Q214	L
1012	1432	1	CON	Q01	A
1013	1433	3	CON	Q03	C
1014	1434	17	CON	Q17	O
1015	1435	14	CON	Q14	L
1016	1436	LOCAL	460	LDI	
1017	1437	1	CON	Q001	GOTO LOCAL
1018	1440	163	GOTO	SNCMD (1456)	
*					
1020	1441	205	CON	Q205	E
1021	1442	24	CON	Q24	T
1022	1443	17	CON	Q17	O
1023	1444	15	CON	Q15	M
1024	1445	5	CON	Q05	E
1025	1446	7327	22	CON	Q22
1026	1447	REMOTE	1	GOSUB	SCHDEV
1026	1450		0		
1027	1451	460	LDI		
1028	1452	222	CON	Q222	SEND REMOTE ENABLE
1029	1453	1	GOSUB	SCMD	
1029	1454		0		
1030	1455	106	C=0	X	
*					
1032	1456	SNCMD	346	CB EX	X
1033	1457		1	GOSUB	SCHDEV
1033	1460		0		
1034	1461		1	GOSUB	LISTEN
1034	1462		0		
1035	1463	306	C=B	X	
1036	1464		1	GOSUB	SCMD
1036	1465	7336	0		
1037	1466	UNLCHK	1	GOSUB	UNL
1037	1467		0		
1038	1470		1	GOLONG	PLERCK
1038	1471		2		

* LISTEN - ADDRESS ANY DEVICE AS A LISTENER *

* THE INTEGER PART OF X IS TAKEN AS THE DEVICE # (31 >= # >=1) *

		ENTRY	LISTNR	
1045				
1046	1472	216	CON	Q216
1047	1473	5	CON	Q05
1048	1474	24	CON	Q24
1049	1475	23	CON	Q23
1050	1476	11	CON	Q11
1051	1477	14	CON	Q14
1052	1500	LISTNR	1	GOSUB
1052	1501		0	
1053	1502	406	A=C	X
1054	1503	460	LDI	
1055	1504	40	CON	32
1056	1505	1406	? A<0	X
1057	1506	LISTER	1	GOLNC
1057	1507		2	
1058	1510	1506	? A#0	X
1059	1511	1753	GONC	LISTER (1506)
1060	1512	246	AC EX	X

IS THE NUMBER < 32 ?
NO, SAY "ADR ERR"

```

1061 1513      1500 HPIL=C 5
1062 1514      1 GOSUB FILEN
1062 1515      0
1063 1516      1 GOSUB PLERCK      CHECK ERROR
1063 1517      0
1064 1520      1 GOLONG LISTEN
1064 1521      2

```

```

*****
INA - INPUT ASCII STRING AND PUT IT TO ALPHA      *
IND - INPUT AN ASCII STRING AS A DECIMAL NUMBER    *
*****

```

```

1071      ENTRY INA
1072      ENTRY INDX
1073      ENTRY INAD2      ** ADD ON JUNE 3,1981
1074      ENTRY INADRD

```

```

1076 1522      204 CON      Q204      D
1077 1523      16 CON      Q16      N
1078 1524 7355 11 CON      Q11      I
1079 1525 INDX 410 SS=      1
1080 1526      116 C=0      W      INITIALIZE DIGIT ENTRY
1081 1527      1156 C=C-1  W
1082 1530      126 C=0      XS
1083 1531      1334 PT=     13
1084 1532      1220 LC      10
1085 1533      1150 REGN=C  9
1086 1534      1 GOSUB STBT10
1086 1535      0
1087 1536      634 PT=     11
1088 1537      1020 LC      8      SAY MANTISSA NON-ZERO
1089 1540      1050 REGN=C  8
1090 1541      123 GOTO     INAD      (1553)

```

```

1092 1542      201 CON      Q201      A
1093 1543      16 CON      Q16      N
1094 1544      11 CON      Q11      I
1095 1545 INA 404 SS=      0
1096 1546      116 C=0      W
1097 1547      460 LDI
1098 1550      27 CON      23      LOAD COUNT FOR 24 CHARS IN ALPHA REG
1099 1551      674 RCR      11
1100 1552      356 BC EX      W      AND SAVE IT IN B.M
1101 1553 INAD 1 GOSUB SCHDEV      GET DEVICE ADDR
1101 1554      0
1102 1555      1704 CLR ST      NOT FOR FIND ID
1103 1556      414 ?SS=1      IND ?
1104 1557      1 GSUBNC CLA      NO, CLEAR ALPHA REG
1104 1560      0
1105 1561      1 GOSUB TALKER
1105 1562      0
1106 1563 INAD2 1670 C=REGN 14      SAVE USER FLAG 17 IN S2
1107 1564      374 RCR      10
1108 1565      776 C=C+C      S
1109 1566      776 C=C+C      S
1110 1567      23 GONC      INAD00 (1571)
1111 1570      1010 S2=      1      FLAG 17 SET(IGNORE CR,LF)
1112 1571 INAD00 1 GOSUB NATNRD      SEND "NAT"
1112 1572      0

```

1113 1573	1104 S9=	0	RESET ERR FLAG IF TIME OUT
1114 1574	33 GOTO	INADRD (1577)	
1115 1575 INAECD	306 C=B	X	
1116 1576	1200 HPIL=C	2	ECHO THE LAST FRAME
1117 1577 INADRD	1 GOSUB	RDDFRM	READ A DATA FRAME
1117 1600	0		
1118 1601	1114 ?S9=1		IS IT A DATA FRAME ?
1119 1602	1 GOLC	INADEX	NO, MIGHT BE AN ETO
1119 1603	3		
1120 1604	1634 PT=	0	
1121 1605	130 G=C		PUT THE BYTE TO G
1122 1606	406 A=C	X	
1123 1607	206 B=A	X	SAVE THE FRAME IN B.X
1124 1610	414 ?S8=1		FOR IND ?
1125 1611	37 GOC	CKCRLF (1614)	YES, CHECK CR/LF
1126 1612	1014 ?S2=1		FLAG 17 SET ?
1127 1613	157 GOC	INAD20 (1630)	YES, IGNORE CR/LF
1128 1614 CKCRLF	460 LDI		
1129 1615	15 CON	13	
1130 1616	1546 ? A#C	X	IS IT A "CR" ?
1131 1617	1563 GONC	INAECD (1575)	YES, IGNORE IT
1132 1620	460 LDI		
1133 1621	12 CON	10	
1134 1622	1546 ? A#C	X	IS IT AN "LF" ?
1135 1623	57 GOC	INAD20 (1630)	NO
1136 1624 INANRD	306 C=B	X	GET LAST DATA FRAME
1137 1625	1 GOSUB	NRDC	SAY NOT READY FOR DATA
1137 1626	0		
1138 1627	753 GOTO	INERCK (1724)	
1139 1630 INAD20	414 ?S8=1		IND ?
1140 1631	227 GOC	IND10 (1653)	YES
1141 1632 INAD22	14 ?S3=1		FIND ID ?
1142 1633	113 GONC	INAD25 (1644)	NO, IS INA
1143 1634	260 C=N		
1144 1635	1434 PT=	1	
1145 1636	1352 ? C#0	WPT	? CHARS YET ?
1146 1637	1367 GOC	INAECD (1575)	YES, IGNORE THE REST
1147 1640	252 AC EX	WPT	
1148 1641	1074 RCR	2	
1149 1642	160 N=C		
1150 1643	1323 GOTO	INAECD (1575)	ECHO AND READ NEXT FRAME
1151 1644 INAD25	1 GOSUB	APPEND	APPEND IT TO ALPHA REG
1151 1645	0		
1152 1646	332 C=B	M	SEE IF ALPHA FULL ?
1153 1647	1172 C=C-1	M	
1154 1650	1547 GOC	INANRD (1624)	ALREADY READ 24 CHARACTERS
1155 1651	372 BC EX	M	
1156 1652 INAD30	1233 GOTO	INAECD (1575)	
1157 1653 IND10	460 LDI		FIND A DIGIT
1158 1654	60 CON	48	ASCII 0
1159 1655	1406 ?A<C	X	IS IT A NO.
1160 1656	117 GOC	ASCDG2 (1667)	SEE IF MINUS SIGN OR D.P.
1161 1657	460 LDI		
1162 1660	72 CON	58	1+ASCII 9
1163 1661	1406 ?A<C	X	IS IT A NO.
1164 1662	223 GONC	ASCDG4 (1704)	SEE IF EEX
1165 1663	460 LDI		YES IT IS A #
1166 1664	40 CON	040	CONVERT TO CN CODE
1167 1665	1106 C=A-C	X	
1168 1666	243 GONC	CHRCON (1712)	INSERT A # AND GET NNTCHP

```

1169 1667 ASCDG2 460 LDI
1170 1670 55 CON 45 ASCII (-) MINUS SIGN
1171 1671 1546 ? A#C X IS IT A MINUS SIGN?
1172 1672 47 GOC ASCDG3 (1676) NO
1173 1673 460 LDI YES
1174 1674 34 CON 034 CN KEY CODE
1175 1675 153 GOTO CHRCON (1712) XEO CHS AND GET NXTCHR
1176 1676 ASCDG3 1046 C=C+1 X C,X = 46
1177 1677 1546 ? A#C X IS IT A "."
1178 1700 1527 GOC INAD30 (1652) NO
1179 1701 460 LDI CONVERT TO CN CODE
1180 1702 32 CON 032
1181 1703 73 GOTO CHRCON (1712) INSERT A "." & GET NXTCHR
1182 1704 ASCDG4 460 LDI
1183 1705 105 CON 69
1184 1706 1546 ? A#C X IS IT EEX
1185 1707 1437 GOC INAD30 (1652) NO, INVALID CHR
1186 1710 460 LDI
1187 1711 33 CON 033
1188 1712 CHRCON 130 G=C PLACE DIGIT INTO G AND PRAY
1189 1713 1 GOSUB DIGENT
1189 1714 0
1190 1715 INVCHR 1353 GOTO INAD30 (1652)

1192 ENTRY INAD30
1193 1716 INAD30 406 A=C X
1194 1717 460 LDI
1195 1720 100 CON 0100
1196 1721 1546 ? A#C X IS IT AN ETO ?
1197 1722 27 GOC INERCK (1724) NO, ERROR
1198 1723 1104 S9= 0
1199 1724 INERCK 414 ?S8=1 IND ?
1200 1725 47 GOC INEK10 (1731) YES
1201 1726 14 ?S3=1 FIND ID ?
1202 1727 1 GOLC UNT YES
1202 1730 3
1203 1731 INEK10 1 GOSUB UNTCHK
1203 1732 0
1204 1733 414 ?S8=1 IND ?
1205 1734 1640 RTN NO
1206 1735 614 ?S11=1 PUSH FLAG SET ?
1207 1736 1 GSUBC R*SUB PUSH STACK IF YES
1207 1737 1
1208 1740 1 GOLONG NOREG9
1209 1741 2
1209 ENTRY R5-R6

1211 1742 R5-R6 544 C=HPIL 5
1211 1743 572
1211 1744 503
1212 1745 1600 HPIL=C 6
1213 1746 1740 RTN

```

```

1215 FILLTO 01746

```

```

*****
*-CX<128= CONVERT X TO BINARY, CHECK X<128

```

*-USES: A(X),C, NO STS, NO PT, 2 ADDITIONAL SUB LEVELS
 *-INPUTS: CONTENTS OF X REG, CHIP 0 ENABLED, HEXMODE
 *-OUTPUTS: A(X)= BINARY NUMBER<128, CHIP 0 ENABLED, HEXMODE

```

1226          ENTRY CX<128
1227 1747 CX<128 1 GOSUB CONV3D      CONVERT X REG TO BINARY
1227 1750          0
1228 1751      406 A=C      X      A= BINARY NUMBER
1229 1752      460 LDI
1230 1753      200 CON      128
1231 1754      1406 ? A<C  X      NUMBER < 128?
1232 1755      1540 RTN C      YES
1233 1756          1 GOLONG ERRDE    NO, DATA ERROR
1233 1757          2
  
```

 *-ACKX= ALPHA CHECK OF X REGISTER (ERRORS IF ALPHA)

*-USES: C
 NO PT, NO STATUS, NO ADDITIONAL SUB LEVELS
 *-INPUTS: CHIP 0 ENABLED, HEX MODE
 *-OUTPUTS: C= X REGISTER, EXCEPT THE SIGN FIELD HAS BEEN DESTROYED

```

1245          ENTRY ACKX
1246          ENTRY ACKC
1247 1760 ACKX 370 C=REGN 3      GET X REGISTER
1248 1761 ACKC 1176 C=C-1 S
1249 1762      1176 C=C-1 S      CHECK FOR ALPHA DATA
1250 1763      1640 RTN NC      NOT ALPHA
1251 1764          1 GOSUB IFC
1251 1765          0
1252 1766          1 GOLONG ERRAD    ERROR= ALPHA DATA
1252 1767          2
  
```

```

1255          ENTRY RENTPH      PATCH OF "RDENT"
  
```

```

738 1257 1770 RENTPH 1 GOSUB CSSTAS 738
1257 1771          0
1258 1772      214 ?S5=1      CASSETTE BUSY ?
1259 1773      1757 GOC      RENTPH (1770) YES, WAIT UNTIL IS NOT BUSY
1260 1774          1 GOSUB CSSTCK  CHECK IF THERE IS A READ ERROR
1260 1775          0
1261 1776          1 GOLONG RDENT 738
1261 1777          2
  
```

```

1264          UNLIST
1267          END
  
```

ERRORS : 0

SYMBOL TABLE

AAU10	212	-	217			
AAU20	220	-	213			
AAU25	222	-	235	216	176	
ACKC	1761	-				
ACKX	1760	-				
AOUT1	1064	-				
AOUT10	1115	-	1172			
AOUT12	1137	-	1134			
AOUT14	1142	-				
AOUT15	1143	-				
AOUT16	1153	-	1147			
AOUT17	1160	-	1152			
AOUT18	1162	-	1127			
AOUT20	1164	-	1161	1140	1124	1101
AOUT30	1203	-	1175			
AOUT31	1222	-	1211	1177		
AOUT32	1224	-	1234	1204	1136	
AOUT33	1232	-	1241			
AOUT34	1233	-				
AOUT35	1235	-	1230			
AOUT40	1242	-	1225			
AOUT50	1246	-	1276			
AOUT55	1254	-	1243			
AOUTFL	1066	-				
AOUTIN	1266	-				
ASCDG2	1667	-	1656			
ASCDG3	1676	-	1672			
ASCDG4	1704	-	1662			
ASP	205	-				
ASP00	224	-	221			
ASP10	231	-	236			
ASP30	237	-	232			
ASP40	241	-				
ASP50	246	-	240			
ASP60	247	-	245			
AUTO10	1373	-				
C=RE	437	-	507	445	435	
CHKLAD	1310	-				
CHRCOM	1712	-	1703	1675	1666	
CKORLE	1614	-	1611			
CSSTA9	700	-				
CSTHOP	142	-				
CXC128	1747	-				
DDT0	320	-				
DSPERJ	1264	-	1334			
DTFLOW	332	-				
FDEV10	607	-	630			
FDEV20	621	-	640			
FDEV36	641	-	667	635		
FDEV38	644	-	633			
FDEV40	652	-	670	647	620	
FDEV50	661	-	627			
FDEV60	664	-	655			
FLAG32	1374	-	1364			
FLNMR	1256	-	1201			
FNDCAS	566	-				

FNDDEV	601	-	574			
FNDPTR	575	-				
FNSTS	702	-				
FNSTSA	703	-	774	701		
FRTR35	631	-	616			
FSP00	1061	-	1054			
FSP04	733	-	731			
FSP06	737	-	734			
FSP10	746	-	720			
FSP30	755	-	761			
FSP45	762	-	756			
FSP48	766	-	1032			
FSP50	767	-	772			
FSP60	1017	-	770			
FSP65	1033	-	1026			
FSP68	1036	-	1024			
FSP70	1040	-	1044			
FSP75	1045	-	1041			
FSP80	1051	-	1013			
FSP85	1053	-	776			
FSPER	773	-	1052	1043	1015	760
FSPEX	775	-	1050			
FSPEX4	777	-	745			
FSPEX6	1012	-	1016			
GETDEV	520	-				
GTD#20	534	-	531			
IDY1ST	175	-	200			
INA	1545	-				
INAD	1553	-	1541			
INAD00	1571	-	1567			
INAD2	1567	-				
INAD20	1630	-	1623	1613		
INAD22	1632	-				
INAD25	1644	-	1633			
INAD30	1652	-	1715	1707	1700	
INADEX	1716	-				
INADRO	1577	-	1574			
INAREO	1575	-	1652	1643	1637	1617
INANRD	1624	-	1650			
IND10	1653	-	1631			
INDX	1525	-				
INEX10	1731	-	1725			
INERCK	1724	-	1722	1627		
INVOHR	1715	-				
LAD	342	-				
LADER	1322	-	1316	1311		
LADERM	1327	-	1323			
LISTEN	335	-				
LISTER	1506	-	1511			
LISTNR	1500	-				
LOCAL	1436	-				
MANIO	1367	-				
NATH10	357	-	363			
NATHRD	352	-				
NRD	364	-				
NRDC	366	-				
NXTD25	550	-				
NXTD30	555	-	552			
NXTDEI	545	-				
NXTDEV	541	-				

00P10	1060	-	1056						
00PCMK	1055	-							
OUTA	1106	-							
PILEN	143	-							
PILPRM	140	-							
R5-R6	1742	-							
RDDFRM	420	-							
RDFM10	421	-	426	424					
RDFM20	430	-	422						
RDFMER	436	-	500	463	460	427	415	362	
RDTPC	466	-							
RDTYPE	464	-							
REMOTE	1447	-							
RENTFH	1770	-	1773						
RSIDY	1410	-							
RTNF+2	656	-	565						
RTPACH	671	-							
RTFHRT	512	-							
RTYP10	473	-	517	513					
RTYP20	474	-	501						
RTYP30	502	-	475						
SCHDEV	1335	-							
SCM0	272	-	345	334	325	322	261	256	
SCM015	275	-							
SCM020	276	-	305						
SCM026	312	-	316						
SCM030	315	-	277						
SCMLER	306	-	317	223					
SDAT10	452	-	457	455					
SDAT20	461	-	453						
SDATA	450	-							
SDATA0	446	-							
SEKSUE	326	-							
SEL010	1320	-	1326						
SELECT	1305	-							
SET10Y	1413	-	1355						
SF51	1345	-							
SF5521	1343	-							
SFRMC	274	-							
SFRTH	1354	-							
SHCMD	1456	-	1440	1430					
SHDATA	346	-							
SRWRT	323	-							
TAD	265	-							
TALKER	262	-							
TRIGER	1426	-							
UNL	257	-							
UNLCHK	1466	-							
UNLRST	443	-							
UNT	254	-							
URTDAT	330	-							

ENTRY TABLE

ACKC	1761	-
ACKX	1760	-
AOUT1	1064	-
AOUTFL	1066	-
AOUTIN	1266	-
ASP	205	-
ASP10	231	-
AUTOIO	1373	-
C=R2	437	-
CHKLRD	1310	-
CSSTAS	700	-
CSTNOF	142	-
CXC128	1747	-
DDT0	326	-
DTFLOW	332	-
FDEV20	621	-
FLNMR	1256	-
FNDURS	566	-
FNDPTR	575	-
FNSTS	702	-
FNSTER	703	-
FS100	1061	-
GETDEV	520	-
INA	1545	-
INAD2	1567	-
INADK	1716	-
INADRD	1577	-
INDX	1525	-
LAD	342	-
LADERM	1327	-
LISTEN	335	-
LISTNR	1500	-
LOCAL	1436	-
MANIO	1367	-
NATNRD	352	-
NRD	364	-
NRDC	366	-
NXTDEI	545	-
NXTDEV	541	-
OPCHK	1055	-
OUTA	1106	-
PILEN	143	-
PILPRM	140	-
R5-R6	1742	-
RDDFRM	420	-
RDFMER	436	-
RDTYPE	466	-
RDTYPE	464	-
REMOTE	1447	-
RENTFH	1779	-
RSIDY	1410	-
RTNP+2	656	-
RTFACH	671	-
RTFHRT	512	-
SCHDEV	1335	-
SCMD	272	-

SCMD15	275	-
SCMD20	276	-
SCMD30	313	-
SCMDER	306	-
SDATA	450	-
SDATA0	446	-
SEKSUE	326	-
SELECT	1305	-
SF51	1345	-
SF5521	1343	-
SFRMC	274	-
SNDATA	346	-
SRWRT	323	-
TAD	265	-
TALKER	262	-
TRIGER	1426	-
UNL	257	-
UNLCHK	1466	-
UNLRSF	443	-
UNT	254	-
URIDAT	330	-

EXTERNAL REFERENCES

ROUTIN	1064	1113	
ROUTIN	1065	1114	
APPEND	1644		
APPEND	1645		
AUTOIO	71		
AUTOIO	70		
CASSET	3		
CASSET	2		
CLA	1557		
CLA	1560		
CONV3D	1305	1500	1747
CONV3D	1306	1501	1750
CREATF	5		
CREATF	4		
CSSTAS	1770		
CSSTAS	1771		
CSSTCK	1774		
CSSTCK	1775		
CSTNOP	65		
CSTNOP	64		
DIGENT	1713		
DIGENT	1714		
DIR	7		
DIR	6		
DSPERR	1264		
DSPERR	1265		
ERRAD	1766		
ERRAD	1767		
ERRDE	1756		
ERRDE	1757		
FINDID	73		
FINDID	72		
FNDPTR	1410		
FNDPTR	1411		
FNSTEA	652		
FNSTEA	653		
GETDEV	605	1341	
GETDEV	606	1342	
GTBYTA	1117		
GTBYTA	1120		
IFC	1764		
IFC	1765		
INA	75		
INA	74		
INACEX	1602		
INACEX	1603		
INCADA	1115		
INCADA	1116		
INDX	75		
INDX	76		
INSTAT	101		
INSTAT	100		
LADERM	1506		
LADERM	1507		
LDSET0	644	1343	
LDSET0	645	1344	

ISTEN	1110	1461	1520		
ISTEN	1111	1462	1521		
ISTNR	103				
ISTNR	102				
LOCAL	105				
LOCAL	104				
IANIO	107				
IANIO	106				
IATNRD	1571				
IATNRD	1572				
IENTAP	11				
IENTAP	10				
IOREG9	1740				
IOREG9	1741				
NRDC	1625				
NRDC	1626				
NXTDEV	625				
NXTDEV	626				
OUTA	111				
OUTA	110				
PILEN	566	577	1335	1514	
PILEN	567	600	1336	1515	
PILPRM	67				
PILPRM	66				
PLERCK	572	1254	1337	1470	1516
PLERCK	573	1255	1340	1471	1517
PLEREX	1256	1327			
PLEREX	1257	1330			
PURGEF	13				
PURGEF	12				
PWRDN	113				
PWRDN	112				
PWRUP	115				
PWRUP	114				
RDDFRM	364	405	1577		
RDDFRM	365	406	1600		
RDENT	1776				
RDENT	1777				
RDTYPE	611				
RDTYPE	612				
READA	15				
READA	14				
READK	17				
READK	16				
READP	21				
READP	20				
READR	23				
READR	22				
READRX	25				
READRX	24				
READS	27				
READS	26				
READSB	31				
READSP	30				
REMOTE	117				
REMOTE	116				
RENAMS	33				
RENAMS	32				
RTRACH	510				
RTRACH	511				

RTFHRT	675			
RTFHRT	676			
R*SUB	1736			
R*SUB	1737			
SCHDEV	1106	1447	1457	1553
SCHDEV	1107	1450	1460	1554
SCMD	350	1453	1464	
SCMD	351	1454	1465	
SDATA	1214	1220		
SDATA	1215	1221		
SDATA0	1162			
SDATA0	1163			
SEC	35			
SEC	34			
SEEKN	326			
SEEKN	327			
SEEKR	37			
SEEKR	36			
SELECT	121			
SELECT	120			
SF51	650			
SF51	651			
SFRMC	373			
SFRMC	374			
SRURT	330			
SRURT	331			
STBT10	1534			
STBT10	1535			
STOPT0	123			
STOPT0	122			
TAD	607			
TAD	610			
TALKER	1561			
TALKER	1562			
TRIGER	125			
TRIGER	124			
UNL	443	1415	1466	
UNL	444	1416	1467	
UNLCHK	1222			
UNLCHK	1223			
UNSEC	41			
UNSEC	40			
UNT	664	1727		
UNT	665	1730		
UNTCHK	1731			
UNTCHK	1732			
VERIFY	43			
VERIFY	42			
URTA	45			
URTA	44			
URTK	47			
URTK	46			
URTP	51			
URTP	50			
URTPV	53			
URTPV	52			
URTR	55			
URTR	54			
URTRX	57			
URTRX	56			

URTS 61
 URTS 60
 ZERO 63
 ZERO 62

End of VASM assembly

VASM ROM ASSEMBLY

REV. 6/81A

OPTIONS: L C S

2		FILE	SCPL2B	
3	0	210 CON	0210	H
4	1	61 CON	061	I
5	2	40 CON	040	
6	3	24 CON	024	T
7	4	23 CON	023	S
8	5	40 CON	040	
9	6	23 CON	023	S
10	7	23 CON	023	S
11	10	1 CON	001	A
12	11	15 CON	015	M
13	12	55 CON	055	-
14				
15		CASSET	ENTRY	CASSET
16				

18	13	NOTAPE	1 GOSUB	PLEREX	
18	14		0		
19	15		16 CON	016	N
20	16		17 CON	017	O
21	17		40 CON	040	
22	20		15 CON	015	M
23	21		5 CON	005	E
24	22		4 CON	004	D
25	23		1015 CON	01015	M
26	24		1 GOLONG	CSEREX	
26	25		2		

*
 *
 * PARWRT - SEND COMMAND - PARTIAL WRITE
 * ASSUME : CASSETTE IS A LISTENER ALREADY
 * RETURN : CASSETTE AS A TALKER
 * USED A,C +1 SUB LEVEL
 *

34		ENTRY	PARWRT
35		ENTRY	SCMDWT

37	26	PARWRT	460 LDI	
38	27		246 CON	0246
39	30	SCMDWT	1 GOSUB	SCMD
39	31		0	

SAD 06 - PARTIAL WRITE

*
 * THE FOLLOWING ROUTINES WILL READ CASSETTE STATUS AND WAIT UNTILL
 * CASSETTE IS DONE WITH LAST OPERATION
 *
 * WAITS - READ CASSETTE STATUS UNTIL IT IS DONE WITH LAST

OPERATION AND CHECK ERROR

USED A,C,S0-7 +1 SUB LEVEL

49		ENTRY	WAITS
50		ENTRY	CSERR
51		ENTRY	CSSTCK
52		ENTRY	TAPERR

54	32	WAITS	1	GOSUB	CSSTAS	READ CASSETTE STATUS
54	33		0			
55	34		1	GOSUB	PLERCK	ERROR CHECK
55	35		0			
56	36		214	785=1		STILL BUSY ?
57	37		1737	GOC	WAITS (32)	YES, LET'S WAIT
58	40	CSSTCK	114	784=1		ANY CASSETTE ERROR ?
59	41		1640	RTN NC		NO

WHEN THERE IS AN ERROR OCCUR, THE LOWER 4 BITS OF THE CASSETTE STATUS IS USED TO PRESENT AN ERROR NUMBER AS FOLLOW :

1	(0001)	- EOT	DRIVE ERROR
2	(0010)	- STALL	DRIVE ERROR
4	(0100)	- DOOR OPEN	NO TAPE
5	(0101)	- NO TAPE	NO TAPE
7	(0111)	- NEW TAPE	NO TAPE
8	(1000)	- TIME OUT	TAPE ERROR
9	(1001)	- RECORD # ERROR	TAPE ERROR
A	(1010)	- CHECKSUM ERROR	TAPE ERROR

72	42	CSERR	14	783=1		TAPE ERROR ?
73	43		137	GOC	TAPERR (56)	YES
74	44		1014	782=1		NO TAPE ?
75	45		1467	GOC	NOTAPE (13)	YES
76	46		1	GOSUB	PLEREX	IS DRIVE ERRORCODE IS 0
76	47		0			
77	50		4	CON	@04	D
78	51		22	CON	@22	R
79	52		11	CON	@11	I
80	53		26	CON	@26	V
81	54		1005	CON	@1005	E
82	55		73	GOTO	DSERJ (64)	
83	56	TAPERF	1	GOSUB	PLEREX	SAY TAPE ERR
83	57		0			
84	60		15	CON	@15	M
85	61		5	CON	@05	E
86	62		4	CON	@04	D
87	63		1015	CON	@1015	M
88	64	DSERJ	1	GOLONG	DEPERF	
88	65		2			

DIR - CASSETTE DIRECTORY FUNCTION

94		ENTRY	DIR
95		ENTRY	DIRROM
96		ENTRY	DIR150

98	66	222	CON	@222	R
99	67	11	CON	@11	I

100	70	4 CON	004	D
101	71 DIR	1 GOSUB	CHKCST	SEE IF CASSETTE THERE & READY
101	72	0		
102	73	1 GOSUB	CLA	CLEAR ALPHA REGISTER
102	74	0		
103	75	1 GOSUB	BLDAPH	PRINT DIRECTORY HEADER
103	76	0		
104	77	116 CON2	4 14	N
105	100	101 CON2	4 1	A
106	101	115 CON2	4 13	M
107	102	105 CON2	4 5	E
108	103	40 CON2	2 0	4 BLANKS
109	104	40 CON2	2 0	
110	105	40 CON2	2 0	
111	106	40 CON2	2 0	
112	107	124 CON2	5 4	T
113	110	131 CON2	5 9	Y
114	111	120 CON2	5 0	P
115	112	105 CON2	4 5	E
116	113	40 CON2	2 0	5 BLANKS
117	114	40 CON2	2 0	
118	115	40 CON2	2 0	
119	116	40 CON2	2 0	
120	117	40 CON2	2 0	
121	120	122 CON2	5 2	R
122	121	105 CON2	4 5	E
123	122	107 CON2	4 7	G
124	123	523 CON	0523	S
125	124	1 GOSUB	PRT11?	SEE IF NEEDS TO PRINT ?
125	125	0		
126	126 DIRROM	1 GOSUB	FNDCAS	LOOK FOR CASSETTE AGAIN
126	127	0		
127	130	1123 GOTO	CSERR (42)	NO CASSETTE FOUND
128	131 DIR20	1 GOSUB	SEEKR2	SEEK & READ REC.2
128	132	0		
129	133 DIR150	1 GOSUB	RENTPH	READ ONE FILE ENTRY
129	134	0		
130	135	630 C=M		GET FILE NAME
131	136	1056 C=C+1	W	REACH END OF DIRECTORY ?
132	137	1 GOLC	UNT	YES, ALL DONE
132	140	3		
133	141	260 C=N		
134	142	1376 ? C#0	S	IS IT A PURGED FILE ?
135	143	1703 GONC	DIR150 (133)	YES, READ NEXT ENTRY
136	144	1 GOSUB	CLA	CLEAR ALPHA REGISTER
136	145	0		
137	146	630 C=M		GET FILE NAME AGAIN
138	147	550 REGN=C	5	PUT NAME TO ALPHA REG.
139	150	1 GOSUB	BLDAPH	PAD IN ONE BLANK
139	151	0		
140	152	440 CON	0440	
*				
142	153 FILTYP	260 C=N		
143	154	106 C=0	X	
144	155	1374 RCR	13	C,X= FILE TYPE
145	156	406 A=C	X	
146	157	1 GOSUB	TYPASC	LOAD THE ASCII
146	160	0		
147	161	10 CON	8	PROG FILE TYPE
148	162	120 CON	0120	P

149 163
 150 164
 151 165
 152 166
 153 167
 154 170
 155 171
 156 172
 157 173
 158 174
 159 175
 160 176
 161 177
 162 200
 163 201
 164 202
 165 203
 166 204
 167 205
 168 206
 169 207
 170 210
 171 211
 172 212
 173 213
 174 214
 175 215
 176 216

522 CON
 303 GOTO
 15 CON
 104 CON
 501 CON
 243 GOTO
 4 CON
 127 CON
 501 CON
 203 GOTO
 5 CON
 113 CON
 505 CON
 143 GOTO
 6 CON
 123 CON
 524 CON
 103 GOTO
 1 CON
 101 CON
 523 CON
 43 GOTO
 0 CON
 77 CON
 477 CON
 260 C=N
 1574 RCR
 1530 ST=C

0522
 FLTYOP (214)
 13
 0104
 0501
 FLTYOP (214)
 4
 0127
 0501
 FLTYOP (214)
 5
 0113
 0505
 FLTYOP (214)
 6
 0123
 0524
 FLTYOP (214)
 1
 0101
 0523
 FLTYOP (214)
 0
 077
 0477
 12

R
 DATA FILE TYPE
 D
 A
 WRITE ALL FILE TYPE
 W
 A
 KEY FILE TYPE
 K
 E
 STATUS FILE
 S
 T
 ASCII DATA FILE
 A
 S
 UNRECOGNIZED FILE
 ?
 ?
 C[1:0] = FILE TYPE

* S3= USER FILE SEQUIRE
 * S1= AUTO RUN, S0= PRIVATE

180 217
 181 220
 182 221 FLTPS
 183 222
 184 223
 185 224
 186 225 FLTP20
 187 226
 188 227
 189 230
 190 231
 191 232
 192 233 FLTP30
 193 234
 194 235
 195 236
 196 237
 197 240
 198 241 FLTP40
 199 242
 200 243
 201 244
 202 245
 203 246
 204 247 PADELK
 205 250
 206 251
 207 252
 208 253

14 253=1
 53 GONC
 1 GOSUB
 0
 54 CON
 523 CON
 1414 251=1
 53 GONC
 1 GOSUB
 0
 54 CON
 501 CON
 1614 250=1
 53 GONC
 1 GOSUB
 0
 54 CON
 520 CON
 404 S8=
 1110 S9=
 1 GOSUB
 0
 1 GOSUB
 0
 770 C=REGN
 1074 RCR
 1434 FT=
 1352 ? C#0
 347 GOC

FLTP20 (225) NO
 BLDAPH
 054
 0523
 FLTP30 (233)
 BLDAPH
 054
 0501
 FLTP40 (241)
 BLDAPH
 054
 0520
 0
 1
 ARGOUT
 DSDLY
 7
 2
 1
 WPT
 PRBIN# (307) YES

COMMA
 S

COMMA
 A

COMMA
 P

DISP FILE NAME & TYPE
 DELAY .6 SECONDS

16 CHARS IN YET ?

204	254	1	GOSUB	BLDAPH	
204	255	0			
205	256	440	CON	0440	
206	257	1703	GOTO	PADBLK (247)	
207			ENTRY	TYPASC	
208	260	TYPASC	660	C=STK	
209	261	TYPASC	1460	CXISA	
210	262		1072	C=C+1	M
211	263		1346	? C#0	X
212	264		103	GONC	BLDAPC (274)
213	265		1546	? A#C	X
214	266		63	GONC	BLDAPC (274)
215	267		1072	C=C+1	M
216	270		1072	C=C+1	M
217	271		1072	C=C+1	M
218				LEGAL	
219	272		1673	GOTO	TYPASC (261)

*
*
* BLDAPH - BUILD ASCII CHARACTER INTO ALPHA REGISTER

* CALLING SEQUENCE:

```

*      GOSUB  BLDAPH
*      CON    (ONE ASCII CODE)
*      CON    .
*      .
*      .
*      CON    00      ( END OF ASCII TABLE)

```

* BLDAPC - SAME AS BLDAPH EXCEPT RETURN ADDR(STK) ALREADY IN C REG
* AS THE FIRST ASCII CHARACTER
* USED A,C,G,PT +1 SUB LEVEL

236		ENTRY	BLDAPH	
237		ENTRY	BLDAPC	
* 				
239	273	BLDAPH	660	C=STK
240	274	BLDAPC	1460	CXISA
241	275		1072	C=C+1
242	276		560	STK=C
243	277		1634	PT=
244	300		130	G=C
245	301		1266	? C#0
246	302		1	GOLC
246	303		3	
247	304		1	GOSUB
247	305		0	
248	306		1653	GOTO

251	307	PRBIN#	1334	PT=	13	
252	310		1020	LC	8	
253	311		436	A=C	5	
254	312		260	C=N		CI3:01= FILE SIZE
255	313		34	PT=	3	
256	314		412	A=C	WPT	= # OF BYTES FOR PROG FILE
257	315		1576	? A#C	5	IS THIS A PROG FILE
258	316		127	GOC	B-DEC (330)	NO
259	317		116	C=0		
260	320		460	LDI		

261	321	7	CON	7
262	322	646	A=A-1	X
263	323	B-RG10	1072 C=C+1	M
264	324	706	A=A-C	X
265	325	1763	GONC	B-RG10 (323)
266	326	74	RCR	3
267	327	406	A=C	X

* B-DEC - CONVERT A BINARY TO A 5 DIGITS DECIMAL NUMBER
 * INPUT A[3:0] = BINARY
 * OUTPUT : A[12:8] = DECIMAL DIGITS
 * USED A,B,C,PT +1 SUB LEVEL

274	330	B-DEC	216 B=A	W	
275	331		256 AC EX	W	
276	332		1 GOSUB	BINBD0	USE "GENNUM" TO DO BINARY-DECIMAL
276	333		0		

* "GENNUM" WILL ONLY CONVERT THE BINARY IN A.X TO DECIMAL. BUT OUR
 * DATA FILE MAY BE LARGER THAN 4096 REGS(HEX 1000), SO WE HAVE TO
 * TAKE CARE THIS CASE.

280	334	106	C=0	X	
281	335	1160	DADD=C		ENABLE CHIP 0 AGAIN
282	336	1334	PT=	13	
283	337	420	LC	4	
284	340	436	A=C	S	
285	341	336	C=B	S	C.S= # OF DIGITS
286	342	1136	C=A-C	S	
287	343	B-D10	1632 A SR	M	SHIFT IN LEADING ZEROS
288	344		1176 C=C-1	S	MAKE THE NUMBER A 5 DIGIT #
289	345	1763	GONC	B-D10 (343)	
290	346	20	LC	0	
291	347	420	LC	4	
292	350	20	LC	0	
293	351	1120	LC	9	
294	352	620	LC	6	C.M= 04096.....
295	353	112	C=0	WPT	
296	354	12	A=0	WPT	
297	355	34	PT=	3	
298	356	302	C=B	PT	GET HIGHEST DIGIT OF ORIGINAL #
299	357	1240	SETDEC		
300	360	B-D20	1142 C=C-1	PT	
301	361	37	GOC	B-D30 (364)	
302	362	532	A=A+C	M	ADD 4096 FOR A HEX.1000
303	363	1753	GONC	B-D20 (360)	
304	364	B-D30	1140 SETHEX		
305	365	1604	S0=	0	INIT LEADING ZERO FLAG
306	366	PRBI10	216 B=A	W	
307	367	460	LDI		
308	370	40	CON	040	LOAD A BLANK
309	371	1634	PT=	0	
310	372	130	G=C		
311	373	1534	PT=	12	
312	374	1302	? B#0	PT	A ZERO ?
313	375	37	GOC	PRBI40 (400)	NO
314	376	1614	?S0=1		LEADING ZERO ?
315	377	63	GONC	PRBI45 (405)	YES
316	400	PRBI40	1610 S0=	1	
317	401	1334	PT=	13	
318	402	316	C=B	W	
319	403	320	LC	3	

320	404	130	G=C		
321	405	1	GOSUB	APPEND	
321	406	0			
322	407	156	AB EX	W	
323	410	1772	A SL	M	
324	411	676	A=A-1	S	OUTPUT 5 DIGIT ALREADY ?
325	412	1543	GONC	PRBI10 (366)	NOT YET
326	413	1	GOSUB	PRT11?	SEE IF NEEDS TO PRINT ?
326	414	0			
327	415	1	GOSUB	FNDCAS	LOOK FOR THE CASSETTE AGAIN
327	416	0			
328	417	0	NOP		
329	420	1	GOSUB	PCHKKB	CHECK KEY BOARD
329	421	0			
330	422	1	GOLONG	DIR150	
330	423	2			

* 332 ENTRY PRT11?

334	424	PRT11?	1	GOSUB	LDSST0	
334	425		0			
335	426		274	ROR	5	
336	427		1730	CST EX		
337	430		14	?S3=1		IN MANUAL MODE ?
338	431		1540	RTN C		YES, DON'T PRINT
339	432		1730	CST EX		
340	433		1614	?S0=1		PRINTER PRESENT ?
341	434		1640	RTN NC		NO,DON'T PRINT
342	435		1545	CON	@1545	GOSUB PRT11
343	436		674	CON	@674	
344	437		1	GOLONG	FLSHRT	PUT RETURN ADDR OF "NFRPU" IN S
344	440		2			
345				FILLTO	@442	
	441		0000	NOP		
	442		0000	NOP		

*
 *

 * SEQ - SECQUIRE A FILE
 * UNSEQ - UNSECQUIRE A FILE

353		ENTRY	SEC	
354		ENTRY	UNSEC	
356	443	203	CON	@203 C
357	444	5	CON	@05 E
358	445	23	CON	@23 S
359	446	1	GOSUB	FLSCHJ
359	447	0		
360	450	410	S8=	1
361	451	113	GOTO	SEQ10 (462)
362	452	203	CON	@203 C
364	453	5	CON	@05 E
365	454	23	CON	@23 S
366	455	16	CON	@16 H
367	456	25	CON	@25 U
368	457	1	GOSUB	FLSCHJ SEARCH THE FILE
369	460	0		

```

369 461      404 S8=      0
370 462 SEQ10 260 C=N
371 463      1574 RCR      12      CI1:0J= FILE TYPE
372 464      1530 ST=C
373 465      4 S3=      0      ASSUME IS UNPROTECT
374 466      414 ?S8=1      PROTECTING ?
375 467      23 GONC      **2      ( 471 ) NO
376 470      10 S3=      1
377 471      1630 C=ST
378 472      1074 RCR      2
379 473      160 N=C
380 474      1076 C=C+1      S      IS A 41C FILE ?
381 475      1 GOLC      FLTYER      NO, DON'T DO SECURE THEN
381 476      3
382 477      333 GOTO      RNAME10 ( 532 ) RESEND THE FILE ENTRY

```

```

*
*****
* RENAME - RENAME A FILE
*****
*

```

		ENTRY	RENAME	
		ENTRY	RNAME10	
391	500	205 CON	@205	E
392	501	15 CON	@15	M
393	502	1 CON	@01	A
394	503	16 CON	@16	N
395	504	5 CON	@05	E
396	505	22 CON	@22	R
397	506	RENAME 1010 S2=	1	SKIP OVER THE OLD NAME IN A-REG
398	507	1 GOSUB	AOUT1	GET THE NEW NAME INTO M
399	510	0		
399	511	76 B=0	S	
400	512	1 GOSUB	FLSCHX	CHECK DUPLICATE FILE NAME
400	513	0		
401	514	630 C=M		
402	515	1356 ? C#0	W	FOUND THE NAME ?
402	516	1 GOLC	DUPFL	YES, SAY "DUP FL NAME"
403	517	3		
404	520	1 GOSUB	FLSCHJ	SEARCH THE FILE
404	521	0		
405	522	1 GOSUB	CHKPCT	CHECK FILE PROTECT
405	523	0		
406	524	1010 S2=	1	
407	525	4 S3=	0	
408	526	1 GOSUB	AOUTFL	GET OLD FILE NAME TO M
408	527	0		
409	530	1 GOSUB	AOUTFL	GET NEW FILE NAME TO M
409	531	0		
410	532	76 B=0	S	DO COPYBF
411	533	1 GOLONG	REWENT	REWRITE THE FILE ENTRY
411	534	2		

```

*
*****
* WRTS - WRITE REGISTERS SEQUENTIALLY
*      WRITE A REGISTERS BLOCK TO A DATA FILE STARTING FROM
*      WHERE THE FILE POINTER IS
* INPUT : X-REG = BBB.EEE
*      WHERE BBB = STARTING REGISTER NUMBER
*      EEE = ENDING REGISTER NUMBER.
*

```


* NO FILE NAME IS REQUIRED, WILL VERIFY THE FILE POINTING *
 * IS A DATA FILE BEFORE THE WRITING *

```

*
424          ENTRY  WRTRX
*
426  535          230 CON    @230          X
427  536          22 CON    @22           R
428  537          24 CON    @24           T
429  540          22 CON    @22           R
430  541          27 CON    @27           W
431  542 WRTRX      1 GOSUB  CHKCST        SEE IF CASSETTE READY
431  543          0
432  544          1 GOSUB  SVBREN        SAVE BYTE # & READ FILE ENTRY
432  545          0
433  546          1 GOSUB  FLSHDT        SEE IF IT IS A DATA FILE
433  547          0
434  550          1 GOSUB  CHKPCT        CHECK IF FILE PROTECTED
434  551          0
435  552          1 GOSUB  DFBDCK        CHECK BOUNDARY
435  553          0
436  554          230 C=G
437  555          1342 ? C#0 PT          WAS LAST TIME A WRITE ?
438  556          177 GOC   WRRS10 ( 575) YES, JUST CONTINUE TO WRITE
439  557          1 GOSUB  SVMODE        SAVE MODE IN BYTE 253
439  560          0
440  561          1170 C=REGN 9          BACK UP ONE RECORD
441  562          1074 RCR    2          C.X=# OF RECORDS PASSED BOF
442  563          416 A=C
443  564          260 C=N
444  565          474 RCR    8          C.X= STARTING RECORD #
445  566          506 A=A+C X          A.X= CURRENT RECORD #
446          LEGAL
447  567          1 GOSUB  SEEK
447  570          0
448  571          1 GOSUB  PARWRT        SEND CMD- PARTIAL WRITE
448  572          0
449  573          1 GOSUB  RSTBP        RESTORE BYTE POINTER
449  574          0
450  575 WRRS10  233 GOTO   SNDRG0 ( 620) GOTO WRITE THE REGISTERS

```

```

* WRTR - WRITE REGISTERS
*   WRITE A BLOCK OF COCONUT REGISTERS TO A FILE STARTING
*   FROM THE BEGINNING OF THE FILE
* INPUT :
*   ALPHA REG - FILE NAME
*   SNDRG0 - SPECIAL ENTRY
*   INPUT : M.X = # OF REGS TO WRITE
*   M15:31 = STARTING REG ADDRESS

```

```

*
463          ENTRY  WRTR
464          ENTRY  WRTRXX
*
465  576          222 CON    @222          R
467  577          24 CON    @24           T
468  600          22 CON    @22           R
469  601 7582      27 CON    @27           W
470  602 WRTR      1 GOSUB  FLSCHD        SEARCH DATA FILE

```

470	603	0			
471	604	1	GOSUB	CHKPCT	CHECK IF THE FILE PROTECTED ?
471	605	0			
472	606	1	GOSUB	SVENTW	SAVE CURRENT FILE ENTRY
472	607	0			
473	610	1	GOSUB	DATALL	CHECK REG & FILE SIZE
473	611	0			
474	612	1	GOSUB	SEEKN	SEEK TO STARTING RECORD
474	613	0			

476			ENTRY	SNDRGS	
477			ENTRY	SNDRGA	
478			ENTRY	SNDR10	
479			ENTRY	SNDRG0	
480			ENTRY	SNDRDN	
481			ENTRY	CSCKUT	

483	614	1	GOSUB	PARWRT	SEND SEC.CMD- PARTIAL WRITE
-----	-----	---	-------	--------	-----------------------------

483	615	0			
484	616	1	GOSUB	LISTEN	

484	617	0			
-----	-----	---	--	--	--

485	620	SNDRG0	404	S8=	0
-----	-----	--------	-----	-----	---

486	621	SNDRGS	1	GOSUB	DTFLOW
-----	-----	--------	---	-------	--------

SEND SEC.CMD- "DATA FOLLOW"

486	622	0			
-----	-----	---	--	--	--

487	623	SNDRGA	144	HPL=CH	1
-----	-----	--------	-----	--------	---

488	624	5	CH=	0001	
-----	-----	---	-----	------	--

WRITE DATA FRAME CONTROL BITS

489	625	SNDR10	630	C=M	
-----	-----	--------	-----	-----	--

GET REG ADDR & REG COUNT

490	626	1146	C=C-1	X	
-----	-----	------	-------	---	--

ALL DONE ?

491	627	177	GOC	SNDR30 (646)	YES
-----	-----	-----	-----	----------------	-----

C.X = TARGET REG ADDR

492	630	74	ROR	3	
-----	-----	----	-----	---	--

493	631	1160	DADD=C		
-----	-----	------	--------	--	--

POINT TO NEXT REG

494	632	1046	C=C+1	X	
-----	-----	------	-------	---	--

495	633	674	ROR	11	
-----	-----	-----	-----	----	--

496	634	530	M=C		
-----	-----	-----	-----	--	--

497	635	70	C=DATA		
-----	-----	----	--------	--	--

498	636	416	A=C	W	
-----	-----	-----	-----	---	--

499	637	456	A=A+B	W	
-----	-----	-----	-------	---	--

500	640	216	B=A	W	
-----	-----	-----	-----	---	--

501	641	1	GOSUB	SNDRG0	
-----	-----	---	-------	--------	--

501	642	0			
-----	-----	---	--	--	--

502	643	1114	?S9=1		ANY ERROR ?
-----	-----	------	-------	--	-------------

503	644	1613	GONC	SNDR10 (625)	NO
-----	-----	------	------	----------------	----

504	645	133	GOTO	WRERCK (660)	
-----	-----	-----	------	----------------	--

505	646	SNDR30	414	?S8=1	
-----	-----	--------	-----	-------	--

506	647	1540	RTN C		
-----	-----	------	-------	--	--

SEND CLOSE RECORD COMMAND

7508 507	650	SNDRDN	460	LDI	
----------	-----	--------	-----	-----	--

508	651	250	CON	0250	
-----	-----	-----	-----	------	--

509	652	1	GOSUB	SCMD	
-----	-----	---	-------	------	--

DDL8

509	653	0			
-----	-----	---	--	--	--

7509 510	654	CSCKUT	1	GOSUB	WAITS
----------	-----	--------	---	-------	-------

CHECK ERROR

510	655	0			
-----	-----	---	--	--	--

511	656	1	GOLONG	UNT	
-----	-----	---	--------	-----	--

511	657	2			
-----	-----	---	--	--	--

512	660	WRERCK	1	GOLONG	CSEKCK
-----	-----	--------	---	--------	--------

ERROR CHECK

512	661	2			
-----	-----	---	--	--	--

 * ZERO - WRITE ZEROS TO A DATA FILE

* ZEROFL - ROUTINE TO WRITE ZEROS TO A FILE

* INPUT : N[7:4] = # OF RECORDS
 * N[3:0] = STARTING RECORD #
 *

			ENTRY	ZERO	
			ENTRY	ZEROFL	
521					
522					
524	662	217	CON	Q217	0
525	663	22	CON	Q22	R
526	664	5	CON	Q65	E
527	665	32	CON	Q32	Z
528	666 ZERO	1	GOSUB	FLSCHD	SEARCH THE DATA FILE
528	667	0			
529	670	1	GOSUB	CHKPCT	CHECK FILE PROTECT
529	671	0			
530	672	1610	S0=	1	REMEMBER LAST OPERATION
531	673	1	GOSUB	SVMODE	WAS A WRITE
531	674	0			
532	675 ZEROFL	1	GOSUB	SEKSUB	SEEK TO THE FILE
532	676	0			
533	677	260	C=N		
534	700	174	RCR	4	C.X = # OF RECORDS IN FILE
535	701	1434	PT=	1	
536	702 ZERO10	1146	C=C-1	X	DONE WITH ALL RECORDS ?
537	703	1517	GOC	CSCKUT (654)	YES
538	704	160	N=C		
539	705	6	A=0	X	
540	706 ZERO20	106	C=0	X	
541	707	1	GOSUB	SDATA0	
541	710	0			
542	711	1114	?S9=1		ANY ERROR ?
543	712	1467	GOC	WRERCK (660)	YES
544	713	552	A=A+1	WPT	WRITE 256 BYTES YET ?
545	714	1723	GONC	ZERO20 (706)	NOT YET
546	715	260	C=N		
547	716	1643	GOTO	ZERO10 (702)	

*
 * SNDRGC - SEND THE CONTENT OF REG.C
 * INPUT : CASSETTE IS IN MIDDLE OF RECEIVING DATA
 * USED A,C +2 SUB LEVEL
 *

			ENTRY	SNDRGC	
553					
555	717 SNDRGC	416	A=C	W	
556	720	1	GOSUB	SDATA	SEND E1&E2(DIGIT 1&0)
556	721	0			
557	722	256	AC EX	W	
558	723	1706	C SR	X	
559	724	1706	C SR	X	
560	725	1374	RCR	13	C[1]= E0, C[0]=MS
561	726	416	A=C	W	
562	727	1	GOSUB	SDATA	SEND E0 & MS
562	730	0			
563	731	1634	PT=	0	SEND 00 FOLLOWED BY MS-M0
564	732 SNDR20	256	AC EX	W	
565	733	1074	RCR	2	
566	734	416	A=C	W	
567	735	1	GOSUB	SDATA	
567	736	0			
568	737	1734	INC PT		
569	740	524	? PT=	6	

```

570 741      1540 RTN C
571 742      1114 ?S9=1
572 743      1673 GONC   SNDR20 ( 732 )
573 744      1740 RTN

```

```

*****
* READRS - READ REGISTERS SEQUENTIALLY
*   READ A DATA FILE STARTING FROM WHERE THE FILE POINTER*
*   IS AND STORE THEM TO 41C DATA REGISTERS DIRECTING BY *
*   X-REG
* INPUT : NO FILE NAME IS REQUIRED, THE FILE HAS TO BE VERFIED *
*   AS A DATA FILE.
*   X-REG = 6BB.EEE
*****

```

```

585          ENTRY  READRX
587 745      230 CON   @230          X
588 746      22 CON   @22           R
589 747      4 CON   @04           D
590 750      1 CON   @01           A
591 751      5 CON   @05           E
592 752      22 CON   @22           R
758593 753 READRX  1 GOSUB  CHKCST
593 754      0
594 755      1 GOSUB  SVBREN      SAVE BYTE # & READ ENTRY
594 756      0
595 757      1 GOSUB  FLSHDT      VERIFY DATA FILE
595 760      0
596 761      1 GOSUB  DFBDCK      CHECK FILE BOUNDARY
596 762      0
597 763      230 C=G
598 764      1342 ? C#0 PT        WAS LAST TIME A WRITE ?
599 765      123 GONC  RDRS10 ( 777 ) NO, LAST TIME WAS A READ TOO
600 766      1604 S0=  0         REMEMBER WAS A READ LAST TIME
601 767      1 GOSUB  SYMODE      SAVE MODE
601 770      0
602 771      1 GOSUB  TALKER
602 772      0
603 773      1 GOSUB  SEEK40      YES, DO A READ
603 774      0
604 775      1 GOSUB  RSTBP       PUT BYTE # BACK
604 776      0
605 777 READRX  1 GOSUB  TALKER
605 1000      0
606 1001      163 GOTO  RDREG0 (1017)

```

```

*****
* READR - READ REGISTERS
*   READ FROM A DATA FILE STARTING FROM THE BEGINNING OF THE*
*   FILE AND STORE THEM TO DATA REGISTERS DIRECTING BY X-REG*
* INPUT : ALPHA REG - FILE NAME
*****

```

```

615          ENTRY  READR
617 1002      222 CON   @222        R
618 1003      4 CON   @04          D
619 1004      1 CON   @01          A
620 1005      5 CON   @05          E

```

621	1006		22	CON	Q22	R
622	1007	READR	1	GOSUB	FLSCHD	CHECK THE DATA FILE
622	1010		0			
623	1011		1	GOSUB	SVENTR	SAVE CURRENT FILE ENTRY
623	1012		0			
624	1013		1	GOSUB	DATALL	COMPUTE # OF REGS TO READ
624	1014		0			
625	1015		1	GOSUB	SEEKRN	SEEK & READ THE 1ST RECORD
625	1016		0			
*						
627				ENTRY	RDREG	READ REGISTERS ROUTINE
628				ENTRY	RDREG0	
629				ENTRY	RDRG10	
630				ENTRY	CSERCK	
*						
760F	632	1017	RDREG0	404	S8=	0
	637	1020	RDREG	1	GOSUB	SNDATA 70E6
	638	1021		0		SEND "SEND DATA"
	634	1022		630	C=M	
	635	1023	RDRG10	1146	C=C-1	X
	636	1024		530	M=C	
	637	1025	RDRG15	630	C=M	
	638	1026		74	RCR	3
	639	1027		1160	DADD=C	
	640	1030		1046	C=C+1	X
	641	1031		674	RCR	11
	642	1032		530	M=C	
	643	1033		1	GOSUB	RDRGA 763E
	643	1034		0		READ 7 BYTE & PUT IT IN A
	644	1035		1360	DATA=C	
	645	1036		456	A=A+B	W
	646	1037		216	B=A	W
	647	1040		674	RCR	11
	648	1041		730	CM EX	
	649	1042		1146	C=C-1	X
	650	1043		47	GOC	RDRG30 (1047)
	651	1044		730	CM EX	YES, IF CARRY
	652	1045		1200	HPIL=C	2
	653	1046		1573	GOTO	RDRG15 (1025)
	654	1047	RDRG30	730	CM EX	
	655	1050		414	?S8=1	GET LAST DATA BYTE BACK
	656	1051		33	GONC	RDRGDN (1054)
	657	1052		1200	HPIL=C	2
	658	1053		1740	RTN	NO
	659	1054	RDRGDN	1	GOSUB	NRDC
	659	1055		0		ECHO
	660	1056		1114	?S9=1	SEND "NOT READY"
	661	1057		57	GOC	CSERCK (1064)
	662	1060		1	GOLONG	UNT
	663	1061		2		
	667	1062	RDRG40	414	?S8=1	ANY ERROR ?
	664	1063		1540	RTN C	YES, SEE WHAT ERROR ?
	665	1064	CSERCK	1104	S9=	0
	666	1065		1	GOSUB	CSSTAS
	666	1066		0		READ CASSETTE STATUS
	667	1067		1114	?S9=1	
	669	1070		27	GOC	LOPERR (1072)
	669	1071		114	?S4=1	
	670	1072	LOPERR	1	GOLNC	PILERR
	670	1073		2		ANY CASSETTE ERROR ?
						NO, MUST BE FIL TRANSMIT ERR

671 1074 1 GOLONG CSERR
671 1075 2

* RDRGA - READ 7 DATA FRAMES AND SAVE IT IN REG-A
* INPUT : CASSETTE IS IN MIDDLE OF READING DATA
* USED A, C, PT +1 SUB LEVEL
*

	ENTRY	RDRGA	
677			
679 1076 RDRGA	1 GOSUB	RDDFRM	READ E2 & E1
679 1077	0		
680 1100	1200 HPIL=C	2	
681 1101	74 RCR	3	
682 1102	416 A=C	W	AC13:113= 0,E1,E2
683 1103	1 GOSUB	RDDFRM	READ E0 & MSFRAME
683 1104	0		
684 1105	1200 HPIL=C	2	
685 1106	1074 RCR	2	CI133= E0
686 1107	436 A=C	S	A.S = E0
687 1110	1074 RCR	2	CI103= MS
688 1111	334 PT=	10	
689 1112	242 AC EX	PT	A= E0,E1,E2,MS
690 1113	1 GOSUB	RDDFRM	READ A DUMMY BYTE
690 1114	0		
691 1115	1200 HPIL=C	2	
692 1116 RDRG50	1 GOSUB	RDDFRM	READ M9-M0
692 1117	0		
693 1120	1114 7S9=1		
694 1121	1417 GOC	RDRG40 (1062)	
695 1122	256 AC EX	W	
696 1123	246 AC EX	X	
697 1124	266 AC EX	XS	
698 1125	1724 DEC PT		
699 1126	224 ? PT=	5	
700 1127	57 GOC	RDRG20 (1134)	
701 1130	1200 HPIL=C	2	ECHO
702 1131	1074 RCR	2	
703 1132	416 A=C	W	
704 1133	1633 GOTO	RDRG50 (1116)	
705 1134 RDRG20	74 RCR	3	
706 1135	416 A=C	W	
707 1136	1740 RTH		

* SEEKR - SET THE FILE POINTER IN A DATA FILE TO A GIVEN REGISTER *
* NUMBER, SO THE FOLLOWING READ OR WRITE REGISTERS SEQUEN- *
* TIALY CAN START FROM THE POINTER (INCLUSIVE). *
* EXAMPLE : SEEK TO REGISTER 23 WILL MAKE THE NEXT READ OR WRITE *
* STARTING FROM REGISTER 23 *
* INPUT : ALPHA REG = FILE NAME *
* X-REG = DESTINATION REGISTER # *

	ENTRY	SEEKR	
719			
721 1137	222 CON	0222	R
722 1140	13 CON	013	K
723 1141	5 CON	005	E
724 1142	5 CON	005	E
725 1143	23 CON	023	S

726	1144	SEEKR	1	GOSUB	FLSCHD 7200	SEARCH THE DATA FILE
726	1145		0			
727	1146		1	GOSUB	SVENTR 7000	SAVE CURRENT FILE ENTRY
727	1147		0			
728	1150		1	GOSUB	X-BIN 7000	CONVERT X TO BINARY
728	1151		0			
729	1152		1	GOSUB	RG-BY# 7000	MULTIPLY REG # BY 7
729	1153		0			
730	1154		1074	ROR	2	
731	1155		416	A=C	W	A[3:0]=DESTINATION REG #
66E 732	1156		260	C=N		C[3:0]= # OF REGS AVAILABLE
733	1157		1156	C=C-1	W	1ST REG IS REG.0
734	1160		34	PT=	3	
735	1161		252	AC EX	WPT	
736	1162		1412	? ACC	WPT	CROSS END OF FILE ?
737	1163		1	GOLC	CSEOF 773K	YES
737	1164		3			
738	1165		256	AC EX		
739	1166		174	ROR	4	C[4:0]= # OF BYTES(DESTINATION)
740	1167		134	PT=	4	
741	1170		412	A=C	WPT	
742	1171		206	B=A	X	SAVE BYTE # IN B.X
743	1172		1616	A SR		
744	1173		1616	A SR		A.X= # OF RECORDS PASSED EOF
745	1174		260	C=N		
746	1175		474	ROR	8	C.X= STARTING RECORD #
747	1176		506	A=A+C	X	A.X= DESTINATION RECORD #
748				LEGAL		
749	1177		1	GOSUB	SEEKRD	SEEK TO RECORD & READ IT
749	1200		0			
750	1201		146	AB EX	X	
751	1202		1	GOSUB	SETBPL	SET BYTE PTR
751	1203		0			
752	1204		1	GOLONG	UNL	
752	1205		2			

 * CREAT - FUNCTION TO CREAT A DATA FILE IN THE CASSETTE *

* INPUT :
 * INT(X) - FILE SIZE IN # OF REGISTERS AND < 99999
 * ALPHA REG - FILE NAME

762		ENTRY	CREATF	
763		ENTRY	DUPFL	
2626 765	1206	205	CON	Q205 E
766	1207	24	CON	Q24 T
767	1210	1	CON	Q01 A
769	1211	5	CON	Q05 E
769	1212	22	CON	Q22 R
779	1213	3	CON	Q03 C
4680 771	1214	136	C=0	S SEARCH FOR DUPLICATED FILE
772	1215	1	GOSUB	FLSCH SEARCH THE FILE
772	1216	0		
773	1217	630	C=M	
774	1220	1356	? C#0	W DUPLICATE FILE NAME ?
775	1221	203	GONG	CRT10 (1241) NO
2692 776	1222	1	GOSUB	FLEREX SAY "DUP FILE NAME"

776	1223	0			
777	1224	4	CON	004	D
778	1225	25	CON	025	U
779	1226	20	CON	020	P
780	1227	40	CON	040	
781	1230	6	CON	006	F
782	1231	14	CON	014	L
783	1232	40	CON	040	
784	1233	16	CON	016	N
785	1234	1	CON	001	A
786	1235	15	CON	015	N
787	1236	1005	CON	01005	E
788	1237	1	GOLONG	CSEREX	
788	1240	2			
789	1241 CRT10	1	GOSUB	CRTFLX	CREAT A FILE BY X
789	1242	0			
790	1243	1	GOLONG	ZEROFL	WRITE ZEROS TO THE FILE
790	1244	2			

* CRTFL - CREAT A FILE

* INPUT : ALPHA REG - FILE NAME
 * C#10:6] - FILE LENGTH IN BYTES
 * C#5:2] - FILE SIZE IN # OF REGS OR # OF BYTES
 * C#1:0] - FILE TYPE
 * CRTFLX - SAME AS CRTFL EXCEPT THE FILE SIZE IS SPECIFIED IN X

* OUTPUT : M= FILE NAME IN ASCII
 * N13:12] = FILE TYPE
 * N11:8] = USER LEVEL FILE SIZE (# OF REGS OR BYTES)
 * N7:4] = FILE SIZE IN # OF RECORDS
 * N3:0] = STARTING RECORD #
 * USED A,B,C,N, S7-3, +2 SUB LEVEL
 * USED REG.9113] & REG.913:0]

806	ENTRY	CRTFL
809	ENTRY	CRTFL0
810	ENTRY	CRTFLX
811	ENTRY	CRF2ND

813	1245 CRTFLX	1	GOSUB	X-BIN	CONVERT X TO BINARY
813	1246	0			
814	1247	1352	? C#0	WPT	X = 0 ?
815	1250	1	GOLNC	X-BER	YES, CAN'T CREAT ZERO REGS
815	1251	2			
816	1252	1	GOSUB	RG-BY#	COMPUYE # OF BYTES
816	1253	0			
817	1254	1434	PT=	1	
818	1255	1520	LC	13	
819	1256 CRTFL0	1634	PT=	0	
820	1257	102	C=0	PT	CI1:0]= FILE TYPE

* NOW CI10:6] = FILE LENGTH IN # OF BYTES
 * CI5:2] = FILE SIZE IN # OF BYTES OR # REGS
 * CI1:0] = FILE TYPE
 * WE WANT TO MOVE B13:10 = CI13:8] & B17:4] = FILE SIZE IN # OF
 * RECORDS

828	1260 CRTFL	574	ROR	6	CI4:0]=LENGTH IN BYTES
829	1261	416	A=C	ALL	AI13:10]=SIZE, AI9:8]=TYPE

830	1262		1074	ROR	2	
831	1263		1434	PT=	1	
832	1264		1512	? A#0	WPT	NEED ANT PARTIAL RECORD ?
837	1265		23	GONC	CRF10 (1267)	NO
838	1266		1056	C=C+1	W	
835	1267	CRF10	406	A=C	X	A.X = # OF RECORDS
836	1270		34	PT=	3	
837	1271		2	A=0	PT	
838	1272		256	AC EX	W	C13:10=SIZE, C9:8=TYPE, C3:0=LENGTH
839	1273		174	ROR	4	
840	1274	CRFL 1	356	BC EX	W	B=LENGTH(4), SIZE(4), TYPE(2), XXXX
841	1275		1	GOSUB	R5-R6 73E2	SAVE 1ST DRIVE ADDR IN R6 7600
841	1276		0			
842	1277	CRF2ND	106	C=0	X	
843	1300		1160	DADD=C		
844	1301		1	GOSUB	SEEKRC	
844	1302		0			
845	1303		1	GOSUB	RDENT	READ 1ST 32 BYTE OF REC.0
845	1304		0			
846	1305		260	C=N		
847	1306		174	ROR	4	C.X= # OF REC. USED FOR DIR
848	1307		34	PT=	3	
849	1310		412	A=C	WPT	A13:03= DIR SIZE IN RECORDS
850	1311		1170	C=REGN	9	
851	1312		252	AC EX	WPT	
852	1313		1046	C=C+1	X	
853	1314		1	GOLC	TAPERR	IF C.X=FFF - PARTIALLY FORMATED TAPE
853	1315		3			
854	1316		1046	C=C+1	X	C13:03= 1ST REC.# AFTER DIR
855	1317		1150	REGN=C	9	ASSUMING DIR STARTS AT REC. #2
856	1320		1146	C=C-1	X	
857	1321		1146	C=C-1	X	
858	1322		746	C=C+C	X	8 ENTRIES PER REC.
859	1323		746	C=C+C	X	
860	1324		746	C=C+C	X	
861	1325		406	A=C	X	A.X= TOTAL # OF ENTRIES
862	1326		1270	C=REGN	10	SAVE THE # IN REG.10
863	1327		246	AC EX	X	
864	1330		1250	REGN=C	10	
865	1331		1	GOSUB	SEEKR2	SEEK TO REC.2 AND READ IT
865	1332		0			
866	1333		46	B=0	X	# OF EXISTING ENTRIES COUNT
867	1334	CRF20	404	S8=	0	
868	1335		146	AB EX	X	CHECK IF ENTRY # >= TOTAL #
869	1336		546	A=A+1	X	
870	1337		1270	C=REGN	10	
871	1340		1406	? A<C	X	
872	1341		253	GONC	TRYNXT (1366)	DIR FULL, TRY NEXT DRIVE
873	1342		146	AB EX	X	
874	1343		1	GOSUB	RENTPH 73EX	READ A FILE ENTRY 76E3
874	1344		0			
875	1345		630	C=M		
876	1346		1056	C=C+1	W	REACH END OF DIRECTORY ?
877	1347		233	GONC	CRF40 (1372)	NOT YET
878	1350		34	PT=	3	
879	1351		1170	C=REGN	9	
880	1352		352	BC EX	WPT	B13:03= STARTING RECORD #
881	1353		316	C=8		
882	1354		406	A=C	X	
883	1355		574	ROR	6	C=TYPE, START #, LENGTH, SIZE

884	1356	160	N=C		CHECK IF ENDING RECORD #		
885	1357	174	RCR	4	WILL OVER END OF TAPE		
886	1360	506	A=A+C	X			
887	1361	460	LDI				
889	1362	1001	CON	513	TOTAL # OF RECORDS IN TAPE		
889	1363	1406	? A<C	X			
890	1364	437	GOC	CRF45 (1427)	NOT OVER END OF TAPE YET		
891	1365	410	S8=	1			
892	1366	1	GOSUB	FNTDEV 7845	LOOK FOR ANOTHER DRIVE		
892	1367	0					
893	1370	453	GOTO	NONXT (1435)	NO ANOTHER DRIVE		
894	1371	1063	GOTO	CRF2ND (1277)	SEARCH ANOTHER DRIVE		
*							
16FA	896	1372	CRF40	260	C=N	COMPUTE END REC. ADDR	
	897	1373		474	RCR	8	CI3:03= STRATING REC. #
	898	1374		416	A=C		
	899	1375		1170	C=REGN	9	
	900	1376		1406	? A<C	X	IS THIS A DUMMY ENTRY ?
	901	1377		1357	GOC	CRF20 (1334)	YES
	902	1400		260	C=N		
	903	1401		174	RCR	4	CI3:03= FILE LENGTH IN RECS.
	904	1402		506	A=A+C	X	A.X= NEXT FILE STARTING REC.#
	905	1403		1170	C=REGN	9	
	906	1404		246	AC EX	X	
	907	1405		1150	REGN=C	9	
	908	1406		260	C=N		
	909	1407		1376	? C#0	S	IS THIS A PURGED FILE ?
	910	1410		1247	GOC	CRF20 (1334)	NO
	911	1411		174	RCR	4	C.X = # OF REC. THIS FILE
	912	1412		416	A=C	W	
	913	1413		316	C=B		C=LENGTH(4),SIZE(4),TYPE(2),XXXX
	914	1414		374	RCR	10	C.X= # OF REC. NEEDED
	915	1415		1406	? A<C	X	BIG ENOUGH ?
	916	1416		1167	GOC	CRF20 (1334)	NO, READ NEXT ENTRY
	917	1417		474	RCR	8	DON'T CHANGE THE OLD LENGTH(RECS)
	918	1420		416	A=C	W	
	919	1421		260	C=N		
	920	1422		1574	RCR	12	
	921	1423		234	PT=	5	
	922	1424		252	AC EX	WF1	ONLY CHANGE TYPE & SIZE
	923	1425		1074	RCR	2	
	924	1426		160	N=C		
7717	925	1427	CRF45	1	GOSUB	AOUTFL 7256	GET FILE NAME INTO M
	925	1430		0			
	926	1431		1	GOSUB	RNAM10 755A	SEND FILE ENTRY
	926	1432		0			
	927	1433		1	GOLONG	FLSHRT	PUT HFR ADDR TO RETURN STACK
	927	1434		2			
7210	928	1435	NONXT	414	?S8=1		DIR FULL ?
	929	1436		103	GONC	DIRFUL (1446)	YES
	930	1437	CSFULL	1	GOSUB	PLEREX	SAY "TAPE FULL"
	930	1440		0			
	931	1441		15	CON	015	M
	932	1442		5	CON	005	E
	933	1443		4	CON	004	D
	934	1444		1015	CON	01015	M
	935	1445		63	GOTO	DSFULL (1453)	
	936	1446	DIRFUL	1	GOSUB	PLEREX	
	936	1447		0			
	937	1450		4	CON	004	D

```

938 1451      11 CON      Q11      I
939 1452    1022 CON      Q1022     R
940 1453 DSFULL      1 GOSUB    MESSL
940 1454      0
941 1455      40 CON      Q40
942 1456      6 CON      Q06      F
943 1457      25 CON      Q25      U
944 1460      14 CON      Q14      L
945 1461    1014 CON      Q1014     L
946 1462      273 GOTO    CSERX1 (1511)

```

```

*
948      ENTRY    LJDLY
949      ENTRY    DSDLY

```

```

*
951 1463 LJDLY      410 98=      1
952 1464      1 GOSUB    MSG105    PRINT ERROR MESSAGE
952 1465      0
953 1466 DSDLY      460 LDI        DELAY .6 SECONDS
954 1467    1777 CON      Q1777
955 1470      746 C=C+C      X
956 1471    1146 C=C-1      X
957 1472    1773 GONC      *-1      (1471)
958 1473    1740 RTN

```

```

*
*
961      ENTRY    CSEOF      SAY "END OF FILE"

```

```

*
962 1474 CSEOF      1 GOSUB    PLEREX
962 1475      0
964 1476      5 CON      Q05      E
965 1477      16 CON      Q16      N
966 1500      4 CON      Q04      D
967 1501      40 CON      Q40
968 1502      17 CON      Q17      O
969 1503      6 CON      Q06      F
970 1504      40 CON      Q40
971 1505      6 CON      Q06      F
972 1506      11 CON      Q11      I
973 1507      14 CON      Q14      L
974 1510    1005 CON      Q1005     E
975 1511 CSERX1    763 GOTO    CSERX2 (1607)

```

LDI
 268
 C=C+C
 C=C-1
 *-1
 RTN

```

*****
*  NEWTAP - INITIALIZE A TAPE (FORMAT)
*****

```

```

*
*  NOTE :
*  IT WILL TAKE 3 MINUTES TO FORMAT A TAPE. SO AFTER INITIATING
*  THE FORMAT FUNCTION, WE WILL NOT WAIT UNTIL THE FORMATTING IS
*  DONE BEFORE WE GIVE THE CONTROL BACK TO MAINFRAME.
*  IF THE FORMATTING HAS NOT BEEN DONE PROPERLY, IT WILL PRESUM-
*  ABLY BE CATCHING LATTER BY SOME OTHER READ OR WRITE FUNCTIONS.

```

```

987      ENTRY    NEWTAP
988 1512      215 CON      Q215      M
989 1513      27 CON      Q27      W
990 1514      405 CON      Q405     E
991 1515      416 CON      Q416     N
992 1516 NEWTAP      0 NGP      NON-PROGRAMMABLE 3 DIGITS OPERANDS
993 1517      32 A=0      M      A.X = REQUIRED ENTRY NUMBER
994 1520      546 A=A+1    X      A.X = ENTRIES # +1

```

995	1521	256	AC EX		
996	1522	756	C=C+C	W	DEVIDE THE # BY 8
997	1523	1474	RCR	1	8 ENTRIES PER REC.
998	1524	1376	? C#0	S	MOD OF 8 = 0 ?
999	1525	23	GONC	NWTP10 (1527)	YES
1000	1526	1046	C=C+1	X	NEED ONE MORE REC.
1001	1527	406	A=C	X	
1002	1530	460	LDI		
1003	1531	71	CON	57	
1004	1532	1406	? A<C	X	X <= 56 ?
1005	1533	1	GOLNC	ERRDE	
1006	1534	2			
1006	1535	206	B=A	X	X SHOULD <= 56 AND > 0
1007	1536	410	S8=	1	
1008	1537	1	GOSUB	CHKCS0	SEE IF CASSETTE PRESENT & READY
1008	1540	0			
1009	1541	1	GOSUB	LISTEN	
1009	1542	0			
1010	1543	460	LDI		
1011	1544	245	CON	@245	SAD 05 LISTEN- FORMAT
1012	1545	1	GOSUB	SCMDWT	SEC.CMD - FORMAT
1012	1546	0			
1013	1547	116	C=0		
1014	1550	134	PT=	4	
1015	1551	220	LC	2	
1016	1552	160	N=C		
1017	1553	1	GOSUB	ZEROFL	WRITE ZERO TO REC. 0&1
1017	1554	0			
* WHEN RETURN FROM "ZEROFL", N = 00000000000000					
**					
1020	1555	1	GOSUB	SEKSUB	SEEK TO REC. 0 & GO INTO WRITE MOD
1020	1556	0			
1021	1557	116	C=0		
1022	1560	1334	PT=	13	WRITE L.I.F. ID TO BYTE 0
1023	1561	1020	LC	8	AND DIR LENGTH TO REC.0
1024	1562	134	PT=	4	
1025	1563	220	LC	2	
1026	1564	120	LC	1	
1027	1565	1334	PT=	13	
1028	1566	1	GOSUB	SNBYTS	
1028	1567	0			
1029	1570	134	PT=	4	
1030	1571	116	C=0		
1031	1572	1	GOSUB	SNBYTS	
1031	1573	0			
1032	1574	306	C=8	X	GET DIR LENGTH FROM B.X
1033	1575	144	HPL=CH	1	
1034	1576	405	CH=	@101	MAKE LAST BYTE AS END FRAME
1035	1577	1	GOSUB	SDATA	
1035	1600	0			
1036	1601	1	GOSUB	WAITS	WAIT UNTIL IT DONE
1036	1602	0			
1037	1603	1	GOSUB	INTDIR	INITIALIZE DIR. BUFR
1037	1604	0			
1038	1605	1	GOLONG	NFRPU	
1038	1606	2			
*					
1040	1607	1	GOLONG	CSEEX	
1040	1610	2			
1041			FILLTO	@1611	

1611

0000 NOP

*
* X-BIN - CONVERT THE INTEGER PART OF X-REG TO BINARY
*

* INPUT : NOTHING

* OUTPUT : C[3:0] = BINARY OF INT(X)

* USED A, B,X, C, P,Q +1 SUB LEVEL

*
1049 ENTRY X-BIN
1050 ENTRY X-BINC
1051 ENTRY X-BER

*
1052 1612 X-BIN 1 GOSUB ACKX CHECK X-REG FOR NUMBER
1053 1613 0
1054 1614 X-BINC 416 A=C W
1055 1615 1526 ? A#0 XS X < 1 ?
1056 1616 23 GONC X-BIN1 (1620) NO
1057 1617 16 A=0 W
1058 1620 X-BIN1 206 B=A X SAVE EXP IN B,X
1059 1621 340 SEL Q
1060 1622 1334 PT= 13
1061 1623 240 SEL P
1062 1624 134 PT= 4
1063 1625 12 A=0 WPT
1064 1626 X-BIN2 1762 A SL PQ SHIFT ONE DIGIT INTO A,S
1065 1627 116 C=0 W
1066 1630 276 AC EX S GET THE DIGIT INTO C,S
1067 1631 1374 RCR 13 C[0] = DIGIT
1068 1632 1012 C=A+C WPT ADD THE DIGIT TO PREVIOUS NUMBER
1069 1633 412 A=C WPT COYP TO A
1070 1634 146 AB EX X GET EXP FROM B,X
1071 1635 346 A=A-1 X DONE WITH CONVERSION ?
1072 1636 1540 RTN C YES
1073 1637 146 AB EX X PUT THE REMAIN EXP BACK TO B,X
1074 1640 752 C=C+C WPT TIMES 10
1075 1641 752 C=C+C WPT
1076 1642 752 C=C+C WPT
1077 1643 1012 C=A+C WPT
1078 1644 512 A=A+C WPT
1079 1645 1613 GONC X-BIN2 (1626)
1080 1646 X-BER 1 GOSUB IFC
1081 1647 0
1082 1650 1 GOLONG ERRDE
1083 1651 2

*
*
*
1085 ENTRY RDLPBK
1086 ENTRY WRLPBK

*
*
1089 1652 RDLPBK 1 GOSUB TALKER
1089 1653 0
1090 1654 460 LDI
1091 1655 301 CON @301
1092 1656 SCMDJ1 1 GOLONG SCMD
1092 1657 2

*
1094 1660 WRLPBK 460 LDI
1095 1661 241 CON @241

-































































97C6

1149	1713	1540	RTN C	YES
1150	1714	344	HPL=CH 3	TURN ON THE PIL CHIP
1151	1715	1	CH= 0000 (0004)	
1152	1716	44	HPL=CH 0	SET MASTER CLEAR
1153	1717	5	CH= 0001 (0005)	
1154	1720	44	HPL=CH 0	
1155	1721	1701	CH= 0360	SET SC=CA=TA=LA=1
1156	1722	144	HPL=CH 1	
1157	1723	1405	CH= 0301	SEND READY FRAME - IDY
1158	1724	460	LDI	
1159	1725	35	CON 29	TRY TO WAKE UP 30 DEVICE
1160	1726	406	A=C X	
1161	1727 WKUP10	460	LDI	
1162	1730	43	CON 35	30 MIL.SEC. FOR EVERY DEVICE
1163	1731	1200	HPIL=C 2	
1164	1732 WKUP20	454	FRAY?	ANY IDY COMES BACK YET ?
1165	1733	67	GOC WKUP30 (1741)	YES
1166	1734	1146	C=C-1 X	TIME OUT FOR 50 MIL.SEC. YET
1167	1735	1753	GONC WKUP20 (1732)	NOT YET
1168	1736	646	A=A-1 X	TRY 30 TIMES YET ?
1169	1737	1703	GONC WKUP10 (1727)	NOT YET
1170	1740	23	GOTO WKUP40 (1742)	
1171	1741 WKUP30	1104	S9= 0	
1172	1742 WKUP40	44	HPL=CH 0	
1173	1743	5	CH= 0001	
1174	1744	1740	RTN	

*
 1176 ENTRY PILERR
 1177 ENTRY PLERCK
 1178 ENTRY ERRRTN
 1179 ENTRY UNTCHK

*
 77E7 1181 1745 UNTCHK 1 GOSUB UNT
 1181 1746 0
 77E7 1182 1747 PLERCK 1114 ?S9=1
 1183 1750 1640 RTN NC
 77E7 1184 1751 PILERR 1 GOSUB PLEREX
 1184 1752 0
 1185 1753 24 CON 024
 1186 1754 22 CON 022
 1187 1755 1 CON 001
 1188 1756 16 CON 016
 1189 1757 23 CON 023
 1190 1760 15 CON 015
 1191 1761 11 CON 011
 1192 1762 24 CON 024
 1193 1763 40 CON 040
 1194 1764 5 CON 005
 1195 1765 22 CON 022
 1196 1766 1022 CON 01022
 1197 1767 ERRRTN 1 GOSUB LEFTJ
 1197 1770 0
 1198 1771 1 GOSUB ENCF00
 1198 1772 0
 1199 1773 1 GOSUB MSGDLY
 1199 1774 0
 1200 1775 1 GOLONG ERR110
 1200 1776 2

T
R
A
N
S
M
I
T

E
R
R

77E7

*
*

*

1204
1207

UNLIST
END

ERRORS : 0

SYMBOL TABLE

B-D10	343	-	345						
B-D20	360	-	363						
B-D30	364	-	361						
B-DEC	330	-	316						
B-RG10	323	-	325						
BLDAPC	274	-	266	264					
BLDAPH	273	-	306						
CASSET	13	-							
CREATF	1214	-							
CRF10	1267	-	1265						
CRF20	1334	-	1416	1410	1377				
CRF2ND	1277	-	1371						
CRF40	1372	-	1347						
CRF45	1427	-	1364						
CRT10	1241	-	1221						
CRTFL	1260	-							
CRTFL0	1256	-							
CRTFLX	1245	-							
CSCOUT	654	-	703						
CSEOF	1474	-							
CSEKCK	1064	-	1057						
CSERR	42	-	130						
CSERX1	1511	-	1462						
CSERX2	1607	-	1511						
CSFULL	1437	-							
CSSTCK	40	-							
DIR	71	-							
DIR150	133	-	143						
DIR20	131	-							
DIRFUL	1446	-	1436						
DIRROM	126	-							
DSDLY	1466	-							
DSEKJ	64	-	55						
DSFULL	1453	-	1445						
DUPFL	1222	-							
ERRRTN	1767	-							
FILTP	153	-							
FLTP20	225	-	220						
FLTP30	237	-	226						
FLTP40	241	-	234						
FLTPS	221	-							
FLTYOP	214	-	210	204	200	174	170	164	
LJDLY	1467	-							
LOPERR	1072	-	1070						
MESSLP	1667	-							
NEWTAP	1516	-							
NONXT	1435	-	1370						
NOTAPE	13	-	45						
NWTP10	1527	-	1525						
PADBLK	247	-	257						
PARWRT	26	-							
PILERR	1751	-							
PLERCK	1747	-							
PLEREX	1667	-							
PRB110	366	-	412						
PRB140	400	-	375						

PRBI45	405	-	377	
PRBIN#	307	-	253	
PRT112	424	-		
PURUP	1700	-		
RDLPBK	1652	-		
RDRG	1020	-		
RDRG0	1017	-	1001	
RDRG10	1023	-		
RDRG15	1025	-	1046	
RDRG20	1134	-	1127	
RDRG30	1047	-	1043	
RDRG40	1062	-	1121	
RDRG50	1116	-	1133	
RDRGA	1076	-		
RDRGDN	1054	-	1051	
RDRS10	777	-	765	
READR	1007	-		
READRX	753	-		
RENAME	506	-		
RNAM10	532	-	477	
SCMDJ1	1656	-	1662	
SCMDMT	30	-		
SEC	446	-		
SEEKR	1144	-		
SEQ10	462	-	451	
SHDR10	625	-	644	
SHDR20	732	-	743	
SHDR30	643	-	627	
SHDRDN	650	-		
SHDRG0	620	-	575	
SHDRGA	623	-		
SHDRG0	717	-		
SHDRG5	621	-		
TAPEER	56	-	43	
TRYNXT	1368	-	1341	
TYPASC	260	-		
TYP510	261	-	272	
UNSEC	457	-		
UNTONK	1745	-		
WAITS	32	-	37	
WKUP10	1727	-	1737	
WKUP20	1732	-	1735	
WKUP30	1741	-	1733	
WKUP40	1742	-	1740	
WKUFLP	1706	-		
WRERCK	660	-	712	645
WRLPBK	1660	-		
WRRS10	575	-	556	
WTR	600	-		
WTRX	542	-		
WTRXX	512	-		
X-BER	1643	-		
X-BIN	1612	-		
X-BIN1	1626	-	1616	
X-BIN2	1626	-	1645	
X-BINC	1614	-		
ZERO	666	-		
ZERG10	702	-	716	
ZERG20	706	-	714	
ZERGFL	675	-		

ENTRY TABLE

BLDAPC	274	-
BLDAPH	273	-
CASSET	13	-
CREATF	1214	-
CRF2ND	1277	-
CRTFL	1260	-
CRTFLO	1256	-
CRTFLX	1245	-
CCKUT	654	-
CSEOF	1474	-
CSEKX	1064	-
CSEPP	42	-
CSSTCK	40	-
DIR	71	-
DIR150	133	-
DIRROM	126	-
DSDLY	1466	-
DUPFL	1222	-
ERRRTH	1767	-
LJDLY	1462	-
MESSLP	1667	-
NEWTAP	1516	-
PARWRT	26	-
PILEER	1751	-
PLERCK	1747	-
PLEREX	1662	-
PRT11?	424	-
PWRUP	1700	-
RDLPEK	1652	-
RDRG	1020	-
RDRG0	1017	-
RDRG10	1023	-
RDRGA	1076	-
READR	1007	-
READRK	753	-
RENAME	506	-
RNAM10	532	-
SCMDWT	30	-
SEC	446	-
SEEKR	1144	-
SHDR10	625	-
SHDRDN	650	-
SHDRG0	620	-
SHDRGA	623	-
SHDRG0	717	-
SHDRG5	621	-
TAPEER	56	-
TYPASC	260	-
UNSEC	452	-
UNTCX	1745	-
WAITS	32	-
WKUFLP	1706	-
WRLPEK	1660	-
WTR	602	-
WTRX	542	-
WTRXX	612	-

X-BER	1646	-
X-BIN	1612	-
X-BINC	1614	-
ZERO	666	-
ZEROFL	675	-

EXTERNAL REFERENCES

ACKX	1612					
ACKX	1613					
ADUTI	507					
ADUTI	510					
ADUTFL	526	530	1427			
ADUTFL	527	531	1430			
APPEND	302	304	405			
APPEND	303	305	406			
ARGOUT	243					
ARGOUT	244					
BINBDC	332					
BINBDC	333					
BLDAPH	75	150	221	227	235	254
BLDAPH	76	151	222	230	236	255
CHKCSO	1537					
CHKCSO	1540					
CHKCST	71	542	753			
CHKCST	72	543	754			
CHKPCT	522	550	604	670		
CHKPCT	523	551	605	671		
CLA	73	144				
CLA	74	145				
CLLCDE	1667					
CLLCDE	1670					
CRTFLX	1241					
CRTFLX	1242					
CSEOF	1163					
CSEOF	1164					
CSERCK	660					
CSERCK	661					
CSEREX	24	1237	1607			
CSEREX	25	1240	1610			
CSERR	1074					
CSERR	1075					
CSSTAS	32	1065				
CSSTAS	33	1066				
DATALL	610	1013				
DATALL	611	1014				
DFBDCX	552	761				
DFBDCX	553	762				
DIR150	422					
DIR150	423					
DSOLY	245					
DSOLY	246					
DSPERR	64					
DSPERR	65					
DTFLOW	621					
DTFLOW	622					
DUPFL	516					
DUPFL	517					
ENCP00	1771					
ENCP00	1772					
ERR110	1775					
ERR110	1776					
ERRDE	1533	1650				
ERRDE	1534	1651				

ERRSUE	1665							
ERRSUE	1666							
FLSCH	1215							
FLSCH	1216							
FLSCHD	602	666	1007	1144				
FLSCHD	603	667	1010	1145				
FLSCHJ	448	457	520					
FLSCHJ	447	460	521					
FLSCHX	512							
FLSCHX	513							
FLSHDT	548	757						
FLSHDT	547	760						
FLCHRT	437	1433						
FLCHRT	440	1434						
FLTYER	475							
FLTYER	476							
FNDCAS	126	415						
FNDCAS	127	416						
FNTDEV	1366							
FNTDEV	1367							
IFC	1646	1663						
IFC	1647	1664						
INTDIR	1603							
INTDIR	1604							
LDSSTO	424							
LDSSTO	425							
LEFTJ	1767							
LEFTJ	1770							
LISTEN	616	1541						
LISTEN	617	1542						
MESSL	1453	1671						
MESSL	1454	1672						
MSGLOS	1464							
MSGLOS	1465							
MSGDLY	1773							
MSGDLY	1774							
HFRPU	1605							
HFRPU	1606							
NRDC	1054							
NRDC	1055							
PARWRT	571	614						
PARWRT	572	615						
PCHKKB	420							
PCHKKB	421							
PILEER	1072							
PILEER	1073							
PLERCK	34	1702						
PLERCK	35	1703						
PLEREX	13	46	56	1222	1437	1446	1474	1751
PLEREX	14	47	57	1223	1440	1447	1475	1752
PRT112	124	413						
PRT112	125	414						
R5-R6	1275							
R5-R6	1276							
RDDFRM	1076	1103	1113	1116				
RDDFRM	1077	1104	1114	1117				
RDEHT	1303							
RDEHT	1304							
RDRGA	1033							
RDRGA	1034							

RENTFH	133	1343		
RENTFH	134	1344		
REMENT	533			
REMENT	534			
RG-BY#	1152	1252		
RG-BY#	1153	1253		
RNAM10	1431			
RNAM10	1432			
RSIDY	1704			
RSIDY	1705			
RSTBP	573	775		
RSTBP	574	776		
SCMD	30	652	1656	
SCMD	31	653	1657	
SCMDUT	1545			
SCMDUT	1546			
SDATA	726	727	735	1577
SDATA	721	730	736	1600
SDATA0	707			
SDATA0	710			
SEEK	567			
SEEK	570			
SEEK40	773			
SEEK40	774			
SEEKN	612			
SEEKN	613			
SEEKR2	131	1331		
SEEKR2	132	1332		
SEEKRC	1301			
SEEKRC	1302			
SEEKRD	1177			
SEEKPD	1200			
SEEKRN	1015			
SEEKRN	1016			
SEKSUP	675	1555		
SEKSUP	676	1556		
SETBPL	1202			
SETBPL	1203			
SNBYTS	1566	1572		
SNBYTS	1567	1573		
SNDATA	1020			
SNDATA	1021			
SNDRGC	641			
SNDRGC	642			
SVSEEN	544	755		
SVSEEN	545	756		
SVENTR	1011	1146		
SVENTR	1012	1147		
SVENTU	606			
SVENTU	607			
SVMODE	557	673	767	
SVMODE	560	674	770	
TALKER	771	777	1652	
TALKER	772	1000	1653	
TAPERP	1314			
TAPERP	1315			
TYPASC	157			
TYPASC	160			
UNL	1204			
UNL	1205			

UNT	137	656	1060	1745
UNT	140	657	1061	1746
WAITS	654	1601		
WAITS	655	1602		
WKUFLP	1700			
WKUFLP	1701			
X-BER	1250			
X-BER	1251			
X-BIN	1150	1245		
X-BIN	1151	1246		
ZERGFL	1243	1553		
ZERGFL	1244	1554		

End of VASM assembly

VASM ROM ASSEMBLY REV. 6/81A

OPTIONS: L C S

2 FILE SCPL3B

*
 * FLSCH - SEARCH FOR A DUPLICATING FILE, WILL GENERATE ERROR MESSAGE
 * IF THE FILE IS FOUND

* INPUT :

* C.S = 0 SEARCH FOR DUPLICATED FILE NAME, ERROR IF FOUND
 * C.S = 8 SEARCH FOR PROGRAM FILE, ERROR IF NOT FOUND
 * C.S =13 SEARCH FOR DATA FILE, ERROR IF NOT FOUND
 * C.S = 4 SEARCH FOR WRITE ALL FILE, ERROR IF NOT FOUND
 * C.S = 5 SEARCH FOR KEY FILE, ERROR IF NOT FOUND
 * C.S = 14 SEARCH A FILE DON'T CARE ITS TYPE, ERROR IF NOT FOUND
 * C.S = 15 SAME AS 14 EXCEPT SEARCHING ALWAYS START FROM RECORD 0
 * EXCEPT FOR C.S=15, ALL THE FILE SEARCHING START FROM DIRECTORY
 * BUFFER
 * S11: S11 IS USED AS A SPECIAL CASE FOR REWRITE PROGRAM. IT IS
 * SET BY THE REGULAR ENTRY. IF C.S = 0 AND S11=0, WILL
 * NOT TREAT THE DUPLICATE FILE AS AN ERROR.

* USED A,B,C,M,N,90-9 +3 SUB LEVEL

* CAUTION : SINCE FLSCH USE +3 SUB LEVEL SO IT WILL LOAD THE
 * ADDR OF NFRPU INTO RETURN STACK BEFORE RETURN

* OUTPUT : IF THE FILE FOUND

* M= FILE NAME
 * N= FILE ENTRY

* IF FILE NOT FOUND WILL RETURN WITH M = 0
 * LEAVE CASSETTE NOT A TALKER NOR A LISTENER

30	ENTRY	FLSCHJ
31	ENTRY	FLSCHI
32	ENTRY	FLSCH
33	ENTRY	FLSCHD
34	ENTRY	FLSCHX
35	ENTRY	FLTYER
36	ENTRY	FLSCHC
37	ENTRY	FLSCHG

39 0 NOCST 1 GO LONG CSNOFD

	39	1	2			
*	41	2	FLSCHI	460	LDI	
	42	3		16	CON	14
	43	4		63	GOTO	FLSCH0 (12)
7805	44	5	FLSCHJ	460	LDI	
	45	6		17	CON	15
	46	7		33	GOTO	FLSCH0 (12)
*	48	10	FLSCHD	460	LDI	DATA FILE TYPE = 13
7808	49	11		15	CON	13
7804	50	12	FLSCH0	1474	RCR	1 C.S = FILE TYPE
7803	51	13	FLSCH	376	BC EX	S
	52	14		1	GOSUB	AOUTIN
	52	15		0		
	53	16	FLSCHX	1	GOSUB	FNDCAS 7476 LOOK FOR CASSETTE — 780E
	53	17		0		
	54	20		1603	GOTO	NOCST (0) CASSETTE NOT FOUND
	55	21		1	GOSUB	R5-R6 7382 SAVE LOOP ADDR IN R6
	55	22		0		
	56	23	FLSCHC	404	S8=	0
	57	24		336	C=B	S
	58	25		1376	? C#0	S NEED TO CHECK NEW TAPE ?
	59	26		33	GONC	FLSH01 (31) NO, START FROM REC.2 ANYWAY
	60	27		1076	C=C+1	S
	61	30		23	GONC	FLSH05 (32)
	62	31	FLSH01	410	S8=	1 DON'T DO COPYEF
	63	32	FLSH05	1	GOSUB	CSRDY SEE IF CASSETTE BUSY
	63	33		0		
	64	34		1704	CLR ST	LEFT SHIFT FILE NAME TO M
	65	35		1	GOSUB	AOUTFL GET FILE NAME FROM A-REG
	65	36		0		
	66	37		630	C=M	
	67	40		1150	REGH=C	9 SAVE FILE NAME IN REG.9
	68	41		336	C=B	S IF B.S=15,0 START FROM REC.00
	69	42		1376	? C#0	S
	70	43		333	GONC	FLSH20 (76)
	71	44		1076	C=C+1	S
	72	45		317	GOC	FLSH20 (76)
	73	46		1	GOSUB	SETBP0 7F36 PREPARE TO SEARCH DIR. BUFF
	73	47		0		
	74	50		1	GOSUB	RDLPBK 74AA TALKER DDT 1 (Send Buffer 1) GO INTO READ LOOP BACK MODE
	74	51		0		
	75	52		6	A=0	X
	76	53	FLSH10	206	B=A	X
	77	54		1	GOSUB	RENT10 78A3 READ NEXT FILE ENTRY
	77	55		0		
	78	56		260	C=M	
	79	57		1376	? C#0	S IS THE FILE PURGED ?
	80	60		103	GONC	FLSH15 (70) YES
	81	61		1170	C=REGN	9 GET FILE NAME
	82	62		416	A=C	
	83	63		630	C=M	
	84	64		1556	? A#C	W IS THIS THE ONE ?
	85	65		603	GONC	FLTYCK (145) YES, WE FOUND IT, CHECK TYPE
	86	66		1056	C=C+1	W REACHED END OF DIR ?
	87	67		77	GOC	FLSH20 (76) YES, LET'S START IT OVER
	88	70	FLSH15	146	AB EX	X GET CURRENT ENTRY #
	89	71		546	A=A+1	X
	90	72		460	LDI	

91	73	10	CON	8	LAST ENTRY # IN BUFR
92	74	1546	? A#C	X	THROUGH DIR BUFR YET ?
93	75	1567	GOC	FLSH10 (53)	NOT YET
94	76	FLSH20	1 GOSUB	SEEKR2	SEEK TO RECORD 2 & READ IT
94	77		0		
95	100	1204	S7=	0	RESET FLAG FOR SEARCH BUFFER
96	101	FLSH30	1 GOSUB	RENTPH	READ ONE DIR. ENTRY
96	102		0		
97	103	FLSH32	260 C=N		
98	104	1376	? C#0	S	PURGED FILE ?
99	105	1743	GONC	FLSH30 (101)	YES
100	106	1170	C=REGN	9	GET TARGET FILE NAME
101	107		416 A=C		
102	110		630 C=M		
103	111	1556	? A#C	W	IS THIS THE FILE ?
104	112	333	GONC	FLTYCK (145)	YES, WE FOUND IT, CHECK TYPE
105	113	1056	C=C+1	W	IS IT END OF DIR. ?
106	114	1653	GONC	FLSH30 (101)	NOT YET
107	115		1 GOSUB	FMTDEV	TRY TO LOOK FOR ANOTHER DRIVE
107	116		0		
108	117	23	GOTO	FLSH35 (121)	DIDN'T FIND NEXT CST
109	120	1033	GOTO	FLSCHC (23)	SEARCH NEXT TAPE
785 110	121	FLSH35	116 C=0		
111	122		530 M=C		
112	123	1336	? B#0	S	LOOKING DUPLICATION ?
113	124	473	GONC	FLSHRT (173)	YES, NOT FOUND IS OK
785 114	125	FLSH36	1 GOSUB	PLEREX 7733	SAY "FL NOT FOUND"
114	126		0		
115	127		6 CON	006	F
116	130		14 CON	014	L
117	131		40 CON	040	
118	132		16 CON	016	N
119	133		17 CON	017	O
120	134		24 CON	024	T
121	135		40 CON	040	
122	136		6 CON	006	F
123	137		17 CON	017	O
124	140		25 CON	025	U
125	141		16 CON	016	N
126	142		1004 CON	01004	D
127	143	FLSHER	1 GOLONG	CSEREX	
127	144		2		
128	145	FLTYCK	336 C=B	S	SEE IF LOOK FOR ANY TYPE ?
129	146		1376 ? C#0	S	B.S = 0 ?
130	147		243 GONC	FLSHRT (173)	YES, LOOK FOR DUPLICATE FILE
131	150		1076 C=C+1	S	B.S = 15 ?
132	151		227 GOC	FLSHRT (173)	YES, ANY TYPE IS OK
133	152		1076 C=C+1	S	B.S = 14 ?
134	153		57 GOC	FLSH70 (160)	YES, DON'T CARE TYPE EITHER
135	154		176 AB EX	S	VERIFY THE FILE TYPE
136	155		260 C=N		
137	156		1576 ? A#C	S	FILE TYPE MATCH ?
138	157		337 GOC	FLTYER (212)	NO, FILE TYPE ERROR
139	160	FLSH70	1214 ?S7=1		FILE FOUND IN BUFFER ?
140	161		127 GOC	FLSHRT (173)	YES, DON'T COPY DIR TO BUFFER
141	162		1 GOSUB	REQADR	SEE IF AT LAST ENTRY OF A RECORD
141	163		0		
142	164		646 A=A-1	X	
143	165		646 A=A-1	X	
144	166		1312 ? B#0	WPT	BYTE POINTER = 0 ?

```

145 167 1 GSUBNC SEEKRD YES, BACK UP 2 REC. & READ IT
146 170 0
146 171 1 GOSUB COPYBF COPY CURRENT DIR REC. TO BUFR
146 172 0
147 ENTRY FLSHRT
148 173 FLSHRT 660 C=STK
149 174 432 A=C M
150 175 116 C=0 W
151 176 1176 C=C-1 S
152 177 1174 RCR 9
153 200 560 STK=C
154 201 272 AC EX M
155 202 740 GOTOC

*
157 ENTRY FLSHDT
*
159 203 FLSHDT 460 LDI SET A.S= 13
160 204 15 CON 13
161 205 1474 RCR 1
162 206 436 A=C S
163 207 260 C=N C.S= FILE TYPE
164 210 1576 ? A#C S IS THIS A DATA FILE ?
165 211 1640 RTN NC YES
166 212 FLTYER 1 GOSUB PLEREX SAY "FL TYPE ERR" 728A
166 213 0
167 214 6 CON 006 F
168 215 14 CON 014 L
169 216 40 CON 040
170 217 24 CON 024 T
171 220 31 CON 031 Y
172 221 20 CON 020 P
173 222 1005 CON 01005 E
174 223 1 GOLONG DSPERR
174 224 2

*
*
* RWCHKA,RWCHKK,RWCHKP - OVER WRITE CHECK
* IF DID NOT FIND A DUPLICATE FILE NAME, RETURN IMMEDIATELY
* IF FOUND A DUPLICATE FILE NAME AND THE FILE TYPE IS RIHGT
* AND THE FILE IS NOT PROTECTED, WILL PURGE THE FILE.
* INPUT C.X = FILE TYPE
* IF M=0 MEANS NO DUPLICATE FILE FOUND
* OTHERWISE, M= DUPLIVATE FILE NAME, N= FILE INFORMATION
*
125 ENTRY RWCHK
*
187 225 RWCHK 1474 RCR 1 7335
188 226 436 A=C S
189 227 630 C=M
190 230 1356 ? C#0 W FOUND DUPLICATE FILE NAME ?
191 231 1640 RTN NC NO
192 232 RWCK20 260 C=N CHECK IF THE SAME TYPE
193 233 1576 ? A#C S SAME TYPE ?
194 234 1 GOLC DUPFL NO, SAY "DUPLICATE FILE"
194 235 3
195 236 236 B=A S GET TYPE BACK
196 237 1 GOLONG PURGE 7651
196 240 2
197 FILLTO 0240
*

```

```

* RIDENT - READ FILE ENTRY
* ASSUME : CASSETTE IS AN ADDRESSED TALKER
* USED H, C, S6, +2 SUB LEVEL
* OUTPUT : M = FILE NAME IN ASCII
*          N = FILE INFORMATION
*          NI(13:12) = FILE TYPE
*              NI(13) = 1 - ASCII DATA FILE, S6=0
*                      = 4 - 41C WRITE ALL FILE, S6=0
*                      = 5 - 41C KEY FILE, S6=0
*                      = 6 - 41C STATUS FILE, S6=0
*                      = 8 - 41C PROGRAM FILE, S6=0
*                      = 13 - FIXED LENGTH(8 BYTES) DATA FILE
*                          S6=0
*                      = 15 - NOT 41C FILE, S6 = 1
*          NI(12) = FILE PROTECT INFORMATION
*          NI(12) = -----
*                  | S | O | A | P |
*                  -----
*          NI(11:8) = FILE STARTING RECORD #
*          NI(7:4) = FILE LENGTH IN # OF RECORDS
*          NI(3:0) = FILE SIZE IN BYTES( PROG FILE ) OR IN REGS
* LEAVE CASSETTE AS A TALKER

```

		ENTRY	RIDENT	
		ENTRY	RENT10	
223				
224				
45A/226	241 RIDENT	1 GOSUB	DDT0	SEND SEC.CMD - SEND DATA
226	242	0		
78A/227	243 RENT10	1 GOSUB	NATNRD	<i>Send SDA and wait for 1. Byte</i>
227	244	0		
228	245	434 PT=	8	
229	246 RENT15	1 GOSUB	RDDFRM	READ 7 BYTES OF NAME
229	247	0		
48A/230	250 RENT10	1200 HPIL=C	2	ECHO
231	251	1 GOSUB	PLERCK	
231	252	0		
232	253	1724 DEC PT		
233	254	1624 ? PT=	0	READ 8 BYTES YET ?
234	255	47 GOC	RENT20 (261)	YES
235	256	1 GOSUB	CK-AX	SAVE THE BYTE IN A
235	257	0		
236	260	1663 GOTO	RENT15 (246)	
237	261 RENT20	256 AC EX	W	
238	262	530 M=C		
239	263 RENT30	1034 PT=	2	SKIP NEXT 2 BYTES AND
240	264	16 A=0		READ 1ST BYTE OF TYPE
241	265	1 GOSUB	SKPFRM	
241	266	0		
242	267	1166 C=C-1	XS	C.XS = F
243	270	1046 C=C+1	X	TEST IF TYPE = -1 ?
244	271	33 GONC	RENT40 (274)	NO
245	272	1156 C=C-1	W	
246	273	530 M=C		N = FFFF..... (END OF DIS)
247	274 RENT40	1 GOSUB	SKPFRM	READ 2ND BYTE OF TYPE
247	275	0		
248	276	460 LDI		
249	277	1 GON	1	SEE IF TYPE = 1 ?
250	300	1546 ? A#C	X	
251	301	37 GOC	RENT45 (304)	NO

252	302	1756	A SL		SHIFT THE TYPE TO RIGHT PLACE
253	303	63	GOTO	RENT50 (311)	
254	304	RENT45	1502 ? A#0	PT	RIGHT TYPE FOR 410 ?
255	305	43	GONC	RENT50 (311)	YES
256	306	460	LDI		UNRECOGNIZED TYPE
257	307	760	CON2	15 0	MAKE IT AN TYPE "F"
258	310	406	A=C	X	
259	311	RENT50	1034 PT=	2	SKIP 2 BYTES AND READ
260	312	1	GOSUB	SKPFRM	1ST BYTE OF START REC.#
260	313	0			
261	314	1	GOSUB	SKPFRM	READ 2ND BYTE OF START #
261	315	0			
262	316	1034	PT=	2	SKIP 2 BYTES AND READ 1ST BYTE
263	317	1	GOSUB	SKPFRM	OF FILE LENGTH
263	320	0			
264	321	1	GOSUB	SKPFRM	READ 2ND BYTE OF FILE LENGTH
264	322	0			
265	323	434	PT=	8	SKIP 8 BYTES & READ 1ST BYTE
266	324	1	GOSUB	SKPFRM	OF FILE SIZE
266	325	0			
267	326	1	GOSUB	SKPFRM	READ 2ND BYTE OF FILE SIZE
267	327	0			
268	330	1	GOSUB	RDDFRM	READ 2ND BYTE OF TYPE
268	331	0			
269	332	1200	HPIL=C	2	
270	333	1074	RCR	2	
271	334	136	C=0	8	C = 0X000... (C[12]=FL OPTION)
272	335	1560	C=CORA		
273	336	160	N=C		
274	337	1	GOLONG	NRD	READ LAST BYTE OF ENTRY
274	340	2			
275			FILLTO	0340	

*
 * SKPFRM - READ A GIVEN NUMBER OF FRAMES WITHOUT STORING THEM
 * SKPFRM FALL INTO CX-AX, SO THE LAST BYTE IS SAVE IN A[1:0]
 * CX-AX - SHIFT A TO LEFT BY ONE BYTE AND SAVE A BYTE IN C[1:0]
 * TO A[1:0]
 *

282		ENTRY	SKPFRM
283		ENTRY	CX-AX
285	341	SKPFRM	1114 ?S9=1
286	342	1540	RTN C
287	343	106	C=0 X
288	344	SKPF10	454 FRAV?
289	345	57	GOC SKPF30 (352)
290	346	1046	C=C+1 X
291	347	27	GOC SKPF20 (351)
292	350	1743	GOTO SKPF10 (344)
293	351	SKPF20	1110 S9= 1
294	352	SKPF30	244 C=HPIL 2
294	353	272	
294	354	203	
295	355	1200	HPIL=C 2
296	356	1624	? PT= 0
297	357	37	GOC CX-AX (362)
298	360	1724	DEC PT
299			LEGAL
300	361	1603	GOTO SKPFRM (341)
301	362	CX-AX	1756 A SL

78F1

```

302 363      1756 A SL
303 364      266 AC EX  XS
304 365      406 A=C   X
305 366      1740 RTN

```

```

*****
* W RTP - WRITE CURRENT PROGRAM
* W RTPP - WRITE CURRENT PROGRAM AS A PRIVATE PROGRAM
*
* INPUT : ALPHA REG - FILE NAME
* ALL THREE FUNCTIONS ARE NON-PROGRAMMABLE FUNCTION, THEY ALL
* REQUIRED TO KEY IN THE PROGRAM NAME WILL BE RECORDED
* TRY TO WRITE A PRGM IN ROM OR TO WRITE A PRIVATE PRGM IS NOT
* ALLOWED
*****

```

```

*
318      ENTRY  W RTP
319      ENTRY  W RTPV
326      ENTRY  W PROM

*
28F 322 367      226 CON      @226      V
323 370      20 CON      @20      P
324 371      24 CON      @24      T
325 372      22 CON      @22      R
326 373      27 CON      @27      W
28C 327 374 W RTPV      610 S11=      1      SET PRIVATE FLAG
328 375      63 GOTO      W RTP00 ( 403 )
329 376      220 CON      @220      P
330 377      24 CON      @24      T
331 400      22 CON      @22      R
332 401      27 CON      @27      W
333 402 W RTP      604 S11=      0      RESET PRIVATE FLAG
334 403 W RTP00      10 S3=      1      GET NAME TO M SHIFT FROM LEFT
335 404      116 C=0
336 405      530 M=C
337 406      1010 S2=      1      SAVE LAST CHAR ADDR IN REG.8
338 407      1 GOSUB      AOUT1
339 410      0
339 411      630 C=M
340 412      1150 REGN=C 9
341 413      1356 ? C#0      LABEL PRESENT ?
342 414      143 GONC      W RTP10 ( 430 ) NO, WRITE CURRENT PROGRAM
343 415      1 GOSUB      ASRCH      SEARCH THE PROGRAM
344 416      0
344 417      1356 ? C#0      W
345 420 W PNFER      1 GOLNC      FLNMR      PROGRAM NOT FOUND
346 421      2
346 422      416 A=C
347 423      1114 ?S9=1      MICRO CODE PRGM ?
348 424      1747 GOC      W PNFER ( 420 ) YES, SAY "NON-EXISTANT"
349 425      1014 ?S2=1      PRGM IN ROM ?
350 426      47 GOC      W PROM ( 432 ) YES, DON'T RECORD IT
351 427      103 GOTO      W RTP20 ( 437 )
352 430 W RTP10      314 ?S10=1      ROM ?
353 431      43 GONC      W RTP15 ( 435 ) NO
354 432 W PROM      1 GOSUB      ERROR
355 433      0
356 434      0 XDEF      MSGROM
357 435 W RTP15      1 GOSUB      GETPC
358 436      0

```

357	437	URTP20	256	AC EX		SAVE PRGM ADDR IN B[7:4]
358	440		374	RCR	10	
359	441		372	BC EX	M	
360	442		214	785=1		FILE NAME FOLLOW ?
361	443		1	GSUBNC	ADUTIN	NO, USE PRGM NAME AS FILE NAME
361	444		0			
362	445		76	B=0	S	
363	446		1	GOSUB	FLSCHX	780E ----- 7926
363	447		0			
364	450		460	LDI		
365	451		10	CON	8	
366	452		1	GOSUB	RMCHK	CHECK DUPLICATE FILE
366	453		0			
367	454		34	PT=	3	
368	455		316	C=B		
369	456		174	RCR	4	
370	457		252	AC EX	WPT	
371	460		1	GOSUB	FLINKA	FIND PRGM END
371	461		0			
372	462		1514	7812=1		PRIVATE ?
373	463		1	GOLC	ERRPR	YES, ERROR
373	464		3			
374	465		416	A=C	W	A[11:8]=PRGM END ADDR
375	466		474	RCR	8	
376	467		412	A=C	WPT	A[3:0]=PRGM END ADDR
377	470		1	GOSUB	CPGMHD	FIND PRGM HEAD
377	471		0			
378	472		1	GOSUB	LDSST0	ENABLE CHIP 0 AND LOAD REG.14
378	473		0			
379	474		674	RCR	11	
380	475		1530	ST=C		S0= USER FLAG 11
381	476		256	C=A	W	C[11:8]=END C[3:0]=HEAD
381	477		416			
382	500		374	RCR	10	
383	501		372	B=C	M	B[7:4] = PRGM HEAD ADDR
383	502		332			
384	503		212	B=A	WPT	B[3:0]=PRGM HEAD ADDR
385	504		1574	RCR	12	C[3:0]=PRGM END ADDR
386	505		412	A=C	WPT	
387	506		1	GOSUB	INCAD2	POINT TO 3RD BYTE OF END
387	507		0			
388	510		152	AB EX	WPT	
389	511		1	GOSUB	CNTBYT	COUNT PRGM LENGTH IN BYTES
389	512		0			
390	513		206	B=A	X	B.X=FILE SIZE IN # OF BYTES
391	514		316	C=B	W	C[7:4]= PROG HEAD ADDR
392	515		374	RCR	10	REG.9[11:8]=PROG HEAD ADDR
393	516		1150	REGN=C	9	REG.9[6:4]=FILE SIZE IN BYTES
394	517		174	RCR	4	C.X= FILE SIZE IN BTES
395	520		132	C=0	M	
396	521		1046	C=C+1	X	
397	522		374	RCR	10	C[8:4]=FILE LENGTH IN BYTES
398	523		246	AC EX	X	
399	524		102	C=0	PT	C[3:0]= FILE SIZE IN BYTES
400	525		1574	RCR	12	
401	526		1434	PT=	1	
402	527		1020	LC	8	C[1]= FILE TYPE
403	530		1614	780=1		WANTED AUTO RUN ?
404	531		33	GONC	WRTP30 (534)	NO
405	532		1042	C=C+1	PT	SET AUTO RUN BIT

```

406 533      1042 C=C+1 PT      CI0]= 2
407 534 WRTP30 614 ?S11=1      WRITE PRIVATE PRGM ?
408 535      23 GONC      WRTP40 ( 537 ) NO
409 536      1042 C=C+1 PT
410      LEGAL
411 537 WRTP40      1 GOSUB CRTFL 7680
411 540      0
796 412 541      1 GOSUB SEKSUB      SEEK & SEND SERIES WRITE
412 542      0
413 543      144 HPL=CH 1
414 544      5 CH= 0001
415 545      1170 C=REGN 9
416 546      174 RCR 4
417 547      160 N=C      N.X = # OF BYTES TO WRITE
418 550      174 RCR 4      CI3:0]=PRGM HEAD ADDR
419 551      34 PT= 3
420 552      412 A=C      WPT
421 553      52 B=0      WPT      B.X = CHECKSUM
422 554 WRTP70 260 C=N
423 555      1146 C=C-1 X      ALL DONE ?
424 556      157 GOC      WRTP80 ( 573 ) YES
425 557      160 N=C
426 560      1 GOSUB NXBYTA
426 561      0
427 562      146 AB EX X
428 563      506 A=A+C X      UPDATE CHECKSUM
429 564      146 AB EX X
430 565      1 GOSUB SDATA      SEND THE BYTE
430 566      0
431 567      1114 ?S9=1      ANY ERROR SO FAR ?
432 570      1643 GONC      WRTP70 ( 554 ) NO
433 571      1 GOLONG PILERR      YES, ABORT
433 572      2
434 573 WRTP80 306 C=B X
435 574      144 HPL=CH 1
436 575      405 CH= 0101      MAKE THE LAST BYTE AN END FRAME
437 576      1 GOSUB SDATA      SEND CHECKSUM BYTES
437 577      0
438 600      1 GOLONG CSCKUT      CHECK STATUS AND UNTALK
438 601      2

```

```

*
*****
* READP -READ PROGRAM FROM A PROGRAM FILE ON THE CASSET *
* READSUB - READ SUBROUTINE. SAME AS READP EXCEPT ALWAYS APPEND *
* THE PROGRAM TO END OF MEMORY *
*
*
* INPUT : ALPHA REG = FILE NAME *
* READP WILL HONOR THE KEY REASSIGNMENT WITH THE ALPHA LABEL, IF *
* THE PROGRAM IS READ IN USER MODE *
*****

```

```

*
450      ENTRY READP
451      ENTRY READSB
*
453 602      202 CON      0202      B
454 603      25 CON      025      U
455 604      23 CON      023      S
456 605      4 CON      004      D
457 606      1 CON      001      A
458 607      5 CON      005      E

```


459	610		22 CON	022	R	
460	611	READSB	604 S11=	0	REMEMBER IS READSUB	
461	612		73 GOTO	RP100 (621)		
*						
463	613		220 CON	0220	P	
464	614		4 CON	004	D	
465	615		1 CON	001	A	
466	616		5 CON	005	E	
467	617		22 CON	022	R	
468	620	READP	610 S11=	1		7990
469	621	RP100	460 LDI			
470	622		10 CON	8		
471	623		1 GOSUB	FLSCH0 180A	LOOK FOR THE PRGM FILE	
471	624		0			
472	625		614 ?S11=1		READSUB ?	
473	626		57 GOC	RP150 (633)	NO, IS READ P	
474	627		1 GOSUB	GTFEND	GET FINAL END ADDR	
474	630		0			
475	631		2 A=0	PT		
476	632		313 GOTO	RP245 (663)		
477	633	RP150	1314 ?S13=1		RUNNING ?	
478	634		57 GOC	RP200 (641)	YES, SEE IF IN LAST PRGM	
479	635		1670 C=REGN	14		
480	636		1530 ST=C			
481	637		114 ?S4=1		SINGLE STEPPING ?	
482	640		173 GONC	RP240 (657)	NO, CHANGE PC	
483	641	RP200	314 ?S10=1		ARE WE IN ROM ?	
484	642		147 GOC	RP220 (656)	YES, DON'T CHANGE PC	
485	643		1 GOSUB	FLINKP	FIND PRGM END	
485	644		0			
486	645		474 RCR	8		
487	646		412 A=C	WPT	A[3:0]=CURRENT PRGM END	
488	647		1 GOSUB	INCADA		
488	650		0			
489	651		1 GOSUB	NXBYTA	GET 3RD BYTE OF "END"	
489	652		0			
490	653		1730 CST EX			
491	654		214 ?S5=1		IS THIS FINAL END ?	
492	655		27 GOC	RP240 (657)	YES, CHANGE PC	
493	656	RP220	604 S11=	0	REMEMBER DON'T CHANGE PC	
494	657	RP240	1 GOSUB	GTFEND	GET FINAL END ADDR	
494	660		0			
495	661		1 GOSUB	CPCM10	GET CURRENT PRGM HEAD	
495	662		0			
496	663	RP245	212 B=A	WPT	B[3:0]=STARTING ADDR	
497	664		1 GOSUB	MEMLFT	COMPUTE AVAILABLE REGS	
497	665		0			
498	666		406 A=C	X	A.X=# OF UNUSED MEM REGS	
499	667		116 C=0	W		
500	670		1160 DADD=C			
501	671		312 C=B	WPT		
502	672		1150 REGN=C	9		
503	673		1570 C=REGN	13		
504	674		246 AC EX	X		
505	675		706 A=A-C	X		
506	676		2 A=0	PT		
507	677		152 AB EX	WPT		
508	700		1 GOSUB	CNTBYT	COMPUTE TOTAL AVAILABLE BYTES	
508	701		0			
509	702		260 C=N		C.X = PRGM SIZE IN BYTES	

510	703	1406	? A<C	X	ENOUGH ROOM ?
511	704	253	GONC	RP250 (731)	
* 513 ENTRY NORMCK					
7365	515	705	NORMCK	1 GOSUB	UNT
	515	706		0	
	516	707		1 GOSUB	LDSST0.
	516	710		0	
	517	711	1314	?S13=1	RUNNING ?
	518	712	47	GOC	NOROOM (716) YES
	519	713	114	?S4=1	SST ?
	520	714	1	GOLNC	PACKE
	520	715	2		NO, PACK AND SAY "TRY AGAIN"
* 522 ENTRY NOROOM					
* 524 716 NOROOM 1 GOSUB PLEREX					
	524	717		0	
	525	720	16	CON	016 N
	526	721	17	CON	017 0
	527	722	40	CON	040
	528	723	22	CON	022 R
	529	724	17	CON	017 0
	530	725	17	CON	017 0
	531	726	1015	CON	01015 M
	532	727	1	GOLONG	CSEREX
	532	730	2		
	533	731	RP250	1 GOSUB	SEEKRN 7F77 SEEK TO THE FILE & READ IT
	533	732		0	
73DB	534	733		1 GOSUB	SNDATA 7086 SEND SEC.CMD- SEND DATA
	534	734		0	
73DD	535	735	1170	C=REGN	9 GET STARTING ADDR
	536	736	16	A=0	W
	537	737	412	A=C	WPT A[3:0]=STARTING ADDR
	538	740	260	C=N	C.X = PROG SIZE IN #K OF BYTES
	539	741	530	M=C	
	540	742	RP300	630	C=M
	541	743	1146	C=C-1	X ALL DONE ?
	542	744	217	GOC	RP320 (765) YES
	543	745	530	M=C	
	544	746	1	GOSUB	RDDFRM READ NEXT FRAME
	544	747		0	
	545	750	1114	?S9=1	ANY ERROR ?
	546	751	277	GOC	RP330 (1000) YES, ABORT
	547	752	1200	HPIL=C	2 ECHO
	548	753	256	AC EX	W
	549	754	174	ROR	4 C.X = RUNNING CHECKSUM
	550	755	1006	C=A+C	X ADD CHECKSUM
	551	756	374	ROR	10
	552	757	256	AC EX	W
	553	760	1	GOSUB	INCADA POINT TO NEXT BYTE
	553	761		0	
	554	762	1	GOSUB	PTBYTA STORE THE BYTE
	554	763		0	
	555	764	1563	GOTO	RP300 (742)
* 557 765 RP320 206 B=A X B.X=LAST BYTE ADDR					
	558	766	1	GOSUB	NRD READ LAST BYTE AS CHECKSUM
	558	767		0	

```

559 770 PP372 256 AC EX W
560 771 174 RCR 4 C.X = CHECKSUM
561 772 126 C=0 XS
562 773 1546 ? A#C X CHECKSUM MATCH ?
563 774 323 GONC RP400 (1026) YES
564 775 1110 S9= 1
565 776 404 S8= 0
566 777 33 GOTO RP335 (1002)
567 1000 RP330 410 S8= 1
568 1001 206 B=A X B.X=LAST BYTE ADDR
ERROR OCCUR, THE PRGM HAS NOT BEEN ALL READ IN MEMORY.
GENERATE AN "END" AT THE BEGINNING OF LOADING SPACE AND CLEAN
MEMORY FROM THE "END" TO LAST BYTE LOADING ADDR.
572 ENTRY RP335
573 1002 RP335 116 C=0
574 1003 1160 DADD=C
575 1004 1170 C=REGN 9 GET STARTING ADDR
576 1005 412 A=C WPT
577 1006 316 C=B SAVE LAST BYTE ADDR IN M
578 1007 530 M=C
579 1010 1 GOSUB CLTAIL CLEAR TRAILING BYTE IN THE REG
579 1011 0
580 1012 246 AC EX X
581 1013 1146 C=C-1 X
582 1014 1160 DADD=C
583 1015 406 A=C X
584 1016 116 C=0 W
585 1017 234 PT= 5
586 1020 1420 LC 12
587 1021 1360 DATA=C
588 1022 630 C=M GET LAST LOADING ADDR
589 1023 246 AC EX X
590 1024 1 GOSUB CLM10 CLEAR THE MEMORY
590 1025 0
591 ENTRY RP400
592 1026 RP400 1 GOSUB .UNT
592 1027 0
593 ENTRY RP401
594 1030 RP401 116 C=0
595 1031 530 M=C
596 1032 1160 DADD=C
597 1033 1170 C=REGN 9 GET STARTING ADDR
598 1034 416 A=C
599 1035 34 PT= 3
600 1036 614 ?S11=1 RUNNING OR SINGLE STEPPING
601 1037 43 GONC RP403 (1043) YES, DON'T CHANGE PC
602 1040 304 S10= 0 CLEAR ROM FLAG
603 1041 1 GOSUB PUTPCX SET PC TO THE BEGINNING OF THIS PROG
603 1042 0
604 1043 RP403 1570 C=REGN 13 GET ADDR OF REG.0
605 1044 74 RCR 3
606 1045 1546 ? A#C X STARTING FROM REG.0 ?
607 1046 163 GONC RP410 (1064) YES
608 1047 1 GOSUB GTBYTA CHANGE PREV. FINAL END
608 1050 0
609 1051 1730 CST EX TO LOCAL END
610 1052 204 S5= 0
611 1053 1730 CST EX
612 1054 1 GOSUB PTBYTA
612 1055 0

```

```

613 1056          1 GOSUB  DECADA          POINT TO 1ST OF PREV. END
613 1057          0
614 1060          1 GOSUB  DECADA
614 1061          0
615 1062        256 AC EX
616 1063        530 M=C                      SAVE PREV. LINK ADDR IN M
617 1064 RP410    116 C=0
618 1065        1160 DADD=C
619 1066        1170 C=REGN  9
620 1067        416 A=C      W
621 1070        1670 C=REGN 14
622 1071        1274 RCR     7                      GET USER MODE FLAG
623 1072        1530 ST=C
624 1073        1204 S7=     0
625 1074        340 SEL Q
626 1075        1334 PT=    13
627 1076        240 SEL P
628 1077        1334 PT=    13
629 1100        1420 LC      12
630 1101        1520 LC      13
631 1102        422 A=C      PQ
632 1103 RP425    34 PT=     3
* START SET LINK HERE - LOOK AT NEXT BYTE SEE IF AN ALBL
*
635 1104 RP430    212 B=A      WPT
636 1105          1 GOSUB  NXBYTA
636 1106          0
637 1107        1074 RCR      2
638 1110        1534 PT=     12
639 1111        1362 ? C#0    PQ                      NULL ?
640 1112        1713 GONC     RP425  (1103) YES
641 1113        1576 ? A#C    S                      AN END OR ALBL ?
642 1114        417 GOC       RP450  (1155) NO
643 1115        1402 ? A<C    PT                      X<>N OR LBL.NN ?
644 1116        377 GOC       RP450  (1155) YES
645 1117          34 PT=     3
* SEE IF IT IS AN END
*
648 1120        212 B=A      WPT
649 1121          1 GOSUB  INCAD2
649 1122          0
650 1123          1 GOSUB  GTBYTA
650 1124          0
651 1125        1574 RCR      12
652 1126        152 AB EX     WPT
653 1127        1042 C=C+1    PT
654 1130        323 GONC     RP470  (1162) IT IS AN END
* RECOMPUTE THE LINK FOR AN ALBL
*
657 1131        630 C=M
658 1132        252 AC EX     WPT
659 1133          1 GOSUB  GENLNK
659 1134          0
660 1135        412 A=C      WPT
661 1136        530 M=C
662 1137        1614 ?S0=1                      ARE WE IN USER MODE ?
663 1140        127 GOC       RP440  (1152) YES, DON'T CLEAR THE KEY CODE
664 1141          1 GOSUB  INCAD2
664 1142          0
665 1143          1 GOSUB  INCADA          POINT TO KEY CODE

```

```

665 1144      0
666 1145      106 C=0      X
667 1146      1 GOSUB   PTBYTA      CLEAR THE KEY CODE
668 1147      0
669 1150      630 C=M
670 1151      412 A=C      WPT
671 1152 RP440  1 GOSUB   DECADA      POINT TO 1 BYTE BEFORE ALEL
672 1153      0
673 1154      33 GOTO    RP460  (1157)
674 1155 RP450  34 PT=      3
675 1156      152 AB EX   WPT
676 1157 RP460  1 GOSUB   NXLDEL      SKPLIN ENTRY NOT CHK ROMFLAG
677 1160      0
678 1161      1233 GOTO    RP430  (1104)
* SET LINK FOR FINAL END AND PUT IT TO PROPER PLACE.
* (FINAL END HAS TO BE RIGHT JUSTIFY IN A REG)
*
679 1162 RP470  202 B=A      PT      SAVE BYTE PTR IN B[3]
680 1163      1 GOSUB   DECADA      POINT BACK ONE BYTE
681 1164      0
682 1165      1 GOSUB   CLTAIL      CLEAR TRAILING BYTE IN REG
683 1166      0
684 1167      142 AB EX   PT      RESTORE BYTE PTR
685 1170      420 LC      4
686 1171      34 PT=      3
687 1172      1402 ? A<C   PT      CAN .END. PUT IN THIS REG ?
688 1173      37 GOC      RP480  (1176) NO
689 1174      402 A=C      PT
690 1175      73 GOTO     RP490  (1204)
691 1176 RP480  246 AC EX   X
692 1177      1146 C=C-1   X      PUT .END. TO NEXT REG
693 1200      1160 DADD=C
694 1201      412 A=C      WPT
695 1202      116 C=0      W
696 1203      1360 DATA=C
* GENERATE LINK FOR FINAL END
*
697 1204 RP490  630 C=M
698 1205      252 AC EX   WPT
699 1206      1 GOSUB   GENLNK
700 1207      0
* UPDATE CHAIN HEAD
*
701 1210      530 M=C      SAVE NEW .END. ADDR IN M
702 1211      1160 DADD=C
703 1212      260 C=N
704 1213      674 RCR      11      C[2:13]=FILE TYPE
705 1214      1530 ST=C
706 1215      70 C=DATA      GIVE LAST BYTE TO FINAL END
707 1216      1434 PT=      1
708 1217      220 LC      2
709 1220      1520 LC      13
* CHECK IF THIS IS A PRIVATE PROGRAM
*
710 1221      114 ?S4=1      PRIVATE PGM ?
711 1222      33 GONC      RP500  (1225) NO
712 1223      1434 PT=      1
713 1224      620 LC      6      SET PRIVATE BIT
* CLEAR JEN BETWEEN OLD FINAL END AND NEW FINAL END
*

```

719	1225	RP500	1360	DATA=C		
720	1226		630	C=M		GET NEW .END. ADDR
721	1227		1	GOSUB	CLNMEM	CLEAN TRAIL MEMORY
721	1230		0			
722	1231		1004	S2=	0	
723	1232		1	GOSUB	RSTKCA	REBUILD KEY REASSIGNMENT
723	1233		0			
724	1234		1114	?S9=1		ANY ERROR ?
725	1235		253	GONC	RP550 (1262)	NO
726	1236		1304	S13=	0	
727	1237		414	?S8=1		CHECKSUM ERROR ?
728	1240		1	GOLC	CSERCK	NO, SEE WHAT ERR IT IS
728	1241		3			
729				ENTRY	CKSUME	
730	1242	CKSUME	1	GOSUB	CKSMER	SAY "CHECKSUM ERR"
730	1243		0			
731	1244		1	GOLONG	CSERE0	
731	1245		2			
732				ENTRY	CKSMER	
734	1246	CKSMER	1	GOSUB	MESSLP	
734	1247		0			
735	1250		22	CON	022	R
736	1251		5	CON	005	E
737	1252		1	CON	001	A
738	1253		4	CON	004	D
739	1254		40	CON	040	
740	1255		5	CON	005	E
741	1256		22	CON	022	R
742	1257		1022	CON	01022	R
743	1260		1	GOLONG	LJDLY	LEFT JUSTIFY & DELAY
743	1261		2			
744	1262	RP550	614	?S11=1		RUNNING OR SST ?
745	1263		43	GONC	RP560 (1267)	YES
746	1264		214	?S5=1		AUTO RUN PROG ?
747	1265		1	GOLC	WKUP80	YES, RUN & SOUND A BEEP
747	1266		3			
748	1267	RP560	1	GOLONG	NFRPU	
748	1270		2			

*
 *
 * CLNMEM - AFTER READ IN A PROGRAM, CLEAN THE SPACE BETWEEN
 * THE OLD FINAL END TO NEW FINAL END.
 * CALL WITH NEW FINAL END ADDR IN C(1210)
 *
 * CLM10 - SPECIAL ENTRY
 * CLEAR MEMORY FROM ADDR IN C.X(LARGER) TO ADDR IN A.X(SMALLER)
 *

738			ENTRY	CLNMEM
739			ENTRY	CLM10

761	1271	CLNMEM	406	A=C	X	
762	1272		106	C=0	X	
763	1273		1160	DADD=C		
764	1274		1570	C=REGN	13	GET OLD FINAL END
765	1275		246	AC EX	X	
766	1276		1550	REGN=C	13	UPDATE NEW FINAL END ADDR
767	1277	CLM10	56	B=0	W	
768	1300	CLM15	1406	? ACC	X	WE DONE ?
769	1301		1640	RTN NC		YES

```

770 1302      1146 C=C-1  X
771 1303      1160 DADD=C
772 1304      356 BC EX  W
773 1305      1360 DATA=C
774 1306      356 BC EX  W
775 1307      1713 GOTO   CLM15 (1300)

```

*

*

* CLTAIL - CLEAR TRAILING BYTE IN A REGISTER
 * INPUT A[3:0]= ADDR ONE BYTE PRECEEDING STARTING BYTE
 * USED C, B[1:0] +1 SUB LEVEL

```

781                      ENTRY  CLTAIL

```

*

```

783 1310 CLTAIL      1 GOSUB  INCADA
783 1311              0
784 1312              106 C=0   X
785 1313              1 GOSUB  PTBYTA
785 1314              0
786 1315              1502 ? A#0 PT
787 1316              1727 GOC   CLTAIL (1310)
788 1317              1740 RTN
789                      FILLTO 01317

```

*

* RSTKCA - REBUILD THE KEY REASSIGNMENT
 * AFTER READ IN THE STATUS TRACKS, IT DESTROYED THE OLD KEY REASSIGN-
 * MENT, SO THE KEY REASSIGNMENT HAS TO BE REBUILT. IN THIS CASE,
 * THE KEY REASSIGNMENT JUST READ IN WILL TAKE PRESIDENT.
 * READ IN A PROGRAM WILL DESTROY THE KEY CODE USED IN LAST PROGRAM.
 * IF READ IN PROGRAM IN USER MODE, THE KEY ASSIGNED TO ANY ALBL IN
 * THE PROGRAM JUST READ IN WANT TO TAKE PRESIDENT. THEREFORE, AFTER
 * READ IN A PROGRAM, THE KEY REASSIGNMENT HAS TO BE REBUILT TOO.
 * THE PROCEDURES ARE AS FOLLOWING:
 * 1. CLEAR ALL THE BIT MAP
 * 2. RESTORE THE KEY REASSIGNMENT OF HARDCODE FUNCTION & XROM FUNCTION
 * 3. RESTORE KEY REASSIGNMENT IN ALBL, IF THE KEY ALREADY BEEN
 * ASSIGNED TO OTHER FUNCTION :
 * A. IF IS AFTER READ IN STATUS TRACK, CLEAR THE KEY CODE IN ALBL.
 * B. IF IS AFTER READ IN A PROGRAM, FIND THE KEY CODE IN SOMEWHERE
 * ELSE AND CLEAR IT THERE.

* CALL WITH S2 = 1 MEANS FROM R/STS
 * = 0 MEANS FROM R/PGM
 *

```

811                      ENTRY  RSTKCA

```

*

```

813 1320 RSTKCA      1 GOSUB  ENCP00
813 1321              0
814 1322              1770 C=REGN 15
815 1323              132 C=0   M          CLEAR BIT MAP
816 1324              136 C=0   S
817 1325              1750 REGN=C 15
818 1326              460 LDI
819 1327              277 CON2   11      15

```

*

* GET THE KEY CODE FROM KEY REASSIGNMENT RECORD AND SET ITS BIT
 * IN BIT MAP.

*

```

824 1330 RTKC10      1046 C=C+1  X
825 1331              1204 S7=   0
826 1332              1250 REGN=C 10

```

823	1333	416	A=C	W	
824	1334	1570	C=REGN	13	
829	1335	256	AC EX	W	
830	1336	1546	? A#C	X	REACH CHAIN HEAD ?
831	1337	313	GONC	RTKC30 (1370)	YES, DONE WITH ALL THOSE REG.S
832	1340	RTKC15	1160	DADD=C	
833	1341	70	C=DATA		
834	1342	416	A=C		SAVE C IN A TEMP
835	1343	106	C=0	X	
836	1344	1160	DADD=C		
837	1345	256	AC EX		RESTORE C
838	1346	1076	C=C+1	S	STILL A KEY REASSIGNMENT REG ?
839	1347	213	GONC	RTKC30 (1370)	NO, DONE WITH ALL THOSE REG
840	1350	1434	PT=	1	
841	1351	1214	?S7=1		FIRST KEY CODE ?
842	1352	23	GONC	*+2 (1354)	YES
843	1353	574	RGR	6	
844	1354	1352	? C#0	WPT	IS THERE A KEY CODE ?
845	1355	63	GONC	RTKC20 (1363)	NO
846	1356	416	A=C		
847	1357	1	GOSUB	TBITMA	
847	1360	0			
848	1361	1	GOSUB	SRBMAP	SET THE BIT IN BIT MAP
848	1362	0			
849	1363	RTKC20	1270	C=REGN	10
850	1364	1214	?S7=1		DONE WITH 1ST KEY CODE ?
851	1365	1437	GOC	RTKC10 (1330)	YES
852	1366	1210	S7=	1	
853	1367	1513	GOTO	RTKC15 (1340)	
854	1370	RTKC30	1014	?S2=1	CALL FROM R/STS ?
855	1371	137	GOC	RTKC40 (1404)	YES
856	1372	1170	C=REGN	9	GET START LOADING ADDR
857	1373	34	PT=	3	
858	1374	412	A=C	WPT	
859	1375	1	GOSUB	DECADA	POINT TO PREV. END ADDR
859	1376	0			
860	1377	1	GOSUB	DECADA	
860	1400	0			
861	1401	1270	C=REGN	10	
862	1402	252	AC EX	WPT	
863	1403	1250	REGN=C	10	SAVE THE ADDR IN REG.10
864	1404	RTKC40	1	GOSUB	GTFEND
864	1405	0			
865	1406	RTKC45	34	PT=	3
866	1407	1	GOSUB	GTLINK	
866	1410	0			
867	1411	1346	? C#0	X	CHAIN END ?
868	1412	1	GOLNC	ENCP00	YES, ALL DONE. RTN
869	1413	2			
869	1414	1	GOSUB	UPLINK	
869	1415	0			
870	1416	1076	C=C+1	S	IS IT AN END ?
871	1417	1673	GONC	RTKC45 (1406)	YES
872	1420	106	C=0	X	
873	1421	1160	DADD=C		
874	1422	252	AC EX	WPT	
875	1423	1150	REGN=C	9	SAVE LINK & ADDR IN REG.9
876	1424	412	A=C	WPT	
877	1425	1	GOSUB	INCAD2	POINT TO KEY CODE OF ALBL
877	1426	0			

878	1427	1	GOSUB	NXBYTA	GET KEY CODE
879	1430	0			
879	1431	252	AC EX	WPT	
880	1432	160	N=C		
881	1433	106	C=0	X	
882	1434	1160	DADD=C		
883	1435	1570	C=REGN	13	
884	1436	530	M=C		
885	1437	26	A=0	XS	
886	1440	1506	? A#0	X	IS THERE A KEY CODE ?
887	1441	323	GONC	RUSR40 (1473)	NO
888	1442	206	B=A	X	
889	1443	1	GOSUB	TBITMA	TEST THE BIT MAP
889	1444	0			
890	1445	1356	? C#0		IS THIS BIT SET ?
891	1446	233	GONC	RUSR30 (1471)	NO, JUST SET IT
892	1447	1014	?S2=1		CALL FROM R/STS ?
893	1450	113	GONC	RUSR25 (1461)	NO
894	1451	260	C=N		GET ADDR WHERE KEY CODE IS
895	1452	416	A=C		
896	1453	106	C=0	X	
897	1454	1	GOSUB	PTBYTA	CLEAR THE KEY CODE
897	1455	0			
898	1456	106	C=0	X	
899	1457	1160	DADD=C		
900	1460	133	GOTO	RUSR40 (1473)	
901	1461	RUSR25 146	AB EX	X	CLEAR KEY CODE SOMEWHERE ELSE
902	1462	1270	C=REGN	10	
903	1463	374	RCR	10	
904	1464	356	BC EX		
905	1465	1410	S1=	1	
906	1466	1	GOSUB	GCPKC0	
906	1467	0			
907	1470	33	GOTO	RUSR40 (1473)	
908	1471	RUSR30 1	GOSUB	SRBMAP	SET BIT MAP
908	1472	0			
909	1473	RUSR40 1170	C=REGN	9	
910	1474	416	A=C		
911	1475	1113	GOTO	RTKC45 (1406)	

*

* WRTS - WRITE STATUS
* STATUS INCLUDES :
* 1. X,Y,Z,T, LASTX AND ALPHA REGISTER
* 2. FLAG 0-43
* 3. SIZE AND SIGMA REGISTER ADDRESS

*

	ENTRY	WRTS	
921			
923	1476	223 CON	@223 S
924	1477	24 CON	@24 T
925	1500	22 CON	@22 R
926	1501	27 CON	@27 W
927	1502	WRTS 136	C=0 S
928	1503	1	GOSUB FLSCH SEARCH FOR DUPLICATE FILE
928	1504	0	
929	1505	460	LDI
930	1506	6	CON 6
931	1507	1	GOSUB RWCHK PURGE IT IF SAME FILE FOUND

931 1510	0		
932 1511	116 C=0		
933 1512	460 LDI		
934 1513	12 CON	10	WRITE 10 REGS START FROM REG 000
935 1514	1 GOSUB	RG-BY#	COMPUTE # OF BYTES NEEDED
935 1515	0		
936 1516	1434 PT=	1	
937 1517	620 LC	6	FILE TYPE
938 1520	1 GOSUB	CRTFL0	CREATE A FILE ENTRY IN DIR
938 1521	0		
939 1522	1 GOSUB	SEKSUB	SEEK TO RECORD & SET WRITE MODE
939 1523	0		
940 1524	1670 C=REGN	14	
941 1525	1150 REGN=C	9	MOVE REG.14 TO REG.9

*
 * COMPUTE SIZE AND STORE IT TO REG.8[13:11]
 * COMPUTE RELATIVE POSITION OF SIGMA REG TP REG0 AND STORE IT
 * TO REG.8[10:8]
 *

947 1526	1 GOSUB	FNDEND	FIND TOP END OF MEM
947 1527	0		
948 1530	116 C=0		
949 1531	1160 DADD=C		
950 1532	1570 C=REGN	13	C.X=REG0, C[8:11]=SIGMA REG
951 1533	74 RCR	3	A.X=# OF DATA REG.(=SIZE)
952 1534	706 A=A-C	X	A.X=REG0
953 1535	256 AC EX	W	C[3:5]=SIZE, A.X=REG0
954 1536	674 RCR	11	A[3:5]=SIZE, A.X=REG0
955 1537	272 AC EX	M	C.X=SIG REG, A.X=REG0
956 1540	474 RCR	8	A.X=SIG REG, C.X=REG0
957 1541	246 AC EX	X	A.X=REL.POS. OF SIG.REG
958 1542	706 A=A-C	X	PUT SIZE & REL.POS. OF SIG.REG
959 1543	234 PT=	5	TO REG.8[13:11] & REG.8[10:8]
960 1544	1070 C=REGN	8	
961 1545	474 RCR	8	
962 1546	252 AC EX	WPT	
963 1547	574 RCR	6	
964 1550	1050 REGN=C	8	
965 1551	116 C=0		
966 1552	460 LDI		
967 1553	12 CON	10	
968 1554	530 M=C		
969 1555	56 B=0		INITIALIZE CHECKSUM
970 1556	1 GOSUB	WRTA20	WRITE 10 REGS & CHECKSUM
970 1557	2		

*

 * READS - READ STATUS FILE
 * NOTE: WILL RESIZE THE MACHINE SO AS WILL WIPE OUT USER RTN STACK*

977	ENTRY	READS	
979 1560	123 CON	0223	S
980 1561	4 CON	004	D
981 1562	1 CON	001	A
982 1563	5 CON	005	E
983 1564	22 CON	022	R
984 1565 READS	460 LDI		
985 1566	6 CON	6	

995	1567	1	GOSUB	FLSCH0	SEARCH THE STATUS FILE
996	1570	0			
997	1571	1	GOSUB	SEEKRN	READ THAT RECORD
997	1572	0			
998	1573	116	C=0		
999	1574	460	LDI		
990	1575	12	CON	10	READ 10 REG AND
991	1576	530	M=C		SAVE THEM IN REG.0-REG.9
992	1577	56	B=0		INITIALIZE CHECKSUM
993	1600	1	GOSUB	RDREG	
993	1601	0			
994	1602	1	GOSUB	RDRGA	READ CHECKSUM
994	1603	0			
995	1604	160	N=C		SAVE CHECKSUM IN N
996	1605	674	RCR	11	
997	1606	1	GOSUB	NRDC	SAY NOT READY FOR DATA
997	1607	0			
998	1610	1	GOSUB	UNT	
998	1611	0			
999	1612	260	C=N		
1000	1613	156	AB EX	W	
1001	1614	1556	? A#C	W	CHECKSUM MATCH ?
1002	1615	1	GOLC	CKSUME	NO, SAY "READ ERR"
1002	1616	3			
* UPDATE FLAGS 0-43, SIZE AND SIGMA REG					
*					
1005	1617	1170	C=REGN	9	LOAD THE FLAGS
1006	1620	416	A=C	W	
1007	1621	1670	C=REGN	14	
1008	1622	272	AC EX	M	
1009	1623	276	AC EX	S	
1010	1624	1650	REGN=C	14	
1011	1625	1070	C=REGN	8	
1012	1626	674	RCR	11	C.X= SIZE
1013	1627	1104	S9=	0	
1014	1630	1	GOSUB	SIZSUB	TRY TO UPDATE SIZE
1014	1631	0			
1015	1632	1114	?S9=1		DID WE MAKE IT ?
1016	1633	113	GONC	RSTS35 (1644)	YES, UPDATE SIGMA REG ADDR
1017	1634	1	GOSUB	PLEREX	
1017	1635	0			
1018	1636	23	CON	023	SAY "SIZE ERR"
1019	1637	11	CON	011	I
1020	1640	32	CON	032	Z
1021	1641	1005	CON	01005	E
1022	1642	1	GOLONG	DSPERR	
1022	1643	2			
1023	1644	RSTS35 1070	C=REGN	8	UPDATE SIGMA REG. ADDR
1024	1645	474	RCR	8	
1025	1646	406	A=C	X	
1026	1647	1570	C=REGN	13	
1027	1650	74	RCR	3	
1028	1651	506	A=A+C	X	
1029	1652	474	RCR	8	
1030	1653	246	AC EX	X	
1031	1654	74	RCR	3	
1032	1655	1550	REGN=C	13	
1033	1656	1740	RTN		
* FNYDEV - TRY TO FIND ANOTHER DRIVE IN THE LOOP					

```

* OUTPUT : RETURN TO P+1 IF ANOTHER DRIVE NOT FOUND
* RETURN TO P+2 IF FOUND ANOTHER DRIVE WITH :
* R5 = NEXT DRIVE ADDRESS
* USED A,C, +2 SUB LEVEL

```

```

* 1041 ENTRY FNTDEV
*
1043 1657 FNTDEV 1670 C=REGN 14
1044 1660 574 RCR 6
1045 1661 776 C=C+C S MANUAL MODE ?
1046 1662 1540 RTN C YES, DON'T SEARCH NEXT DRIVE
1047 1663 SKPDEV 1 GOSUB ASP
1047 1664 0
1048 1665 674 RCR 11
1049 1666 432 A=C M
1050 1667 36 A=0 S
1051 1670 576 A=A+1 S
1052 LEGAL
1053 1671 1 GOSUB FDEV20
1053 1672 0
1054 1673 1740 RTN
1055 1674 1014 ?S2=1 NEW TAPE OR NO TAPE ?
1056 1675 33 GONC FNDNXT (1700) NO, FOUND NEXT DEIVR
1057 1676 1414 ?S1=1 NO TAPE ?
1058 1677 1643 GONC SKPDEV (1663) YES, SKIP THIS DRIVE
1059 1700 FNDNXT 1 GOLONG RTNP+2
1059 1701 2

```

```

*****
* INSTAT - READ A BYTE OF DEVICE STATUS
* THE BYTE WILL BE STORED IN USER FLAG 0-7 (F0 IS LSB), AND
* LOWER 6 BITS OF THE BYTE WILL BE CONVERT INTO A DECIMAL
* NUMBER AND STORED IN X-REG.
* IF MORE THAN ONE BYTE ARE READ ONLY STORE THE 1ST NON-ZERO*
* BYTE.
* THE DEVICE HAS TO RESPOND TO RDY FRAME "SSP", OTHERWISE,
* WILL RETURN WITH ZERO.
*****

```

```

* 1072 ENTRY INSTAT
*
1074 1702 224 CON @224 T
1075 1703 1 CON @01 A
1076 1704 24 CON @24 T
1077 1705 23 CON @23 S
1078 1706 16 CON @16 N
1079 1707 11 CON @11 I
1080 1710 INSTAT 1 GOSUB SCHDEV GET DEVICE ADDRESS
1080 1711 0
1081 1712 1 GOSUB TALKER MAKE IT AS A TALKER
1081 1713 0
1082 1714 460 LDI
1083 1715 141 CON @141 SEND RDY FRAME -"SSP"
1084 1716 1 GOSUB RDTYPC
1084 1717 0
1085 1720 460 LDI
1086 1721 1777 CON @1777
1087 1722 1660 C=C,A MASK OFF TOP BITS OF THE STATUS B
1088 1723 346 BC EX X
1089 1724 1646 B SR X SAVE LOWER 6 BITS IN B

```

1090	1725	1	GOSUB	UNTCHK	CHECK ERROR
1090	1726	0			
1091	1727	1134	PT=	9	REVERSE THE ORDER OF THE 8 BITS
1092	1730	246	AC EX	X	C[2:1]=REMAINDING STATUS BYTE
1093	1731	26	A=0	XS	
1094	1732	746	C=C+C	X	LEFT SHIFT ONE BIT TO C.XS
1095	1733	23	GONC	STS20 (1735)	TOP BIT IS ZERO
1096	1734	566	A=A+1	XS	A.XS = 1
1097	1735	246	AC EX	X	
1098	1736	746	C=C+C	X	RIGHT SHIFT THE BIT INTO C[1:0]
1099	1737	746	C=C+C	X	
1100	1740	746	C=C+C	X	
1101	1741	1474	ROR	1	
1102	1742	246	AC EX	X	
1103	1743	1724	DEC PT		
1104	1744	1424	? PT=	1	
1105	1745	1643	GONC	STS10 (1731)	
1106	1746	1670	C=REGN	14	
1107	1747	1574	ROR	12	PUT THE BYTE TO REG.14[13:12]
1108	1750	252	AC EX	WPT	
1109	1751	1074	ROR	2	
1110	1752	1650	REGN=C	14	
1111	1753	1	GOSUB	ANNOUT	
1111	1754	0			
1112	1755	1	GOLONG	FNID60	PUT LOWER 6 BITS TO X-REG
1112	1756	2			
1113			FILLTO	@1760	
	1757	0000	NOP		
	1760	0000	NOP		

*
 *
 * RG-BY# - CONVERT # OF REGISTERS INTO # OF BYTES (MULTIPLY THE
 * REGISTER # BY 8)
 * INPIT : C[3:0] = # OF REGISTERS
 * OUTPUT : C[10:6] = # OF BYTES
 * C[5:2] = # OF REGS OR BYTES
 * A[4:0] = # OF BYTES
 * PT = 4
 * USED A,C,PT +0 SUB LEVEL
 *

1125		ENTRY	RG-BY#
------	--	-------	--------

BF1

1126	1761	RG-BY#	134	PT=	4	
1129	1762		102	C=0	PT	
1130	1763		412	A=C	WPT	A.WPT = FILE SIZE IN REGS
1131	1764		752	C=C+C	WPT	CONVERT # OF REGS INTO # OF BYTES
1132	1765		752	C=C+C	WPT	
1133	1766		752	C=C+C	WPT	TIMES 8
1134	1767		374	ROR	10	C[8:4] = # OF BYTES
1135	1770		34	PT=	3	
1136	1771		252	AC EX	WPT	C[3:0] = # OF REGS
1137	1772		174	ROR	4	C[4:0] = # BYTES
1138	1773		416	A=C	W	A[4:0] = # OF BYTES
1139	1774		474	ROR	8	C[10:6]=BYTES, C[5:2]=REGS
1140	1775		134	PT=	4	
1141	1776		1740	RTN		

*
 *
 *

*

1148
1149

UNLIST
END

ERRORS : 0

SYMBOL TABLE

CKSMER	1246	-			
CKSUMS	1242	-			
CLM10	1277	-			
CLM15	1300	-	1307		
CLNMEN	1271	-			
CLTAIL	1310	-	1316		
CX-AX	362	-	357		
FLSCH	13	-			
FLSCH0	12	-	7	4	
FLSCHC	23	-	120		
FLSCHD	10	-			
FLSCH1	2	-			
FLSCHJ	5	-			
FLSCHW	16	-			
FLSH01	31	-	26		
FLSH05	32	-	30		
FLSH10	53	-	75		
FLSH15	70	-	60		
FLSH20	76	-	67	45	43
FLSH30	101	-	114	105	
FLSH32	103	-			
FLSH35	121	-	117		
FLSH70	160	-	153		
FLSHDT	203	-			
FLSHER	143	-			
FLSHRT	173	-	161	151	147 124
FLTYCK	145	-	112	65	
FLTYER	212	-	157		
FNDNXT	1700	-	1675		
FNTDEV	1657	-			
INSTAT	1710	-			
NOCST	0	-	20		
NORMCK	705	-			
NOPOOM	716	-	712		
RDENT	241	-			
READP	620	-			
READS	1565	-			
READSP	611	-			
RENT10	243	-			
RENT15	246	-	260		
RENT20	261	-	255		
RENT30	262	-			
RENT40	274	-	271		
RENT45	304	-	301		
RENT50	311	-	305	303	
RQ-BY%	1761	-			
RP100	621	-	612		
RP150	633	-	626		
RP200	641	-	634		
RP220	656	-	642		
RP240	657	-	655	640	
RP245	667	-	632		
RP250	731	-	704		
RP300	742	-	764		
RP320	765	-	744		
RP330	1000	-	751		

RP335	1002	-	777	
RP400	1026	-	774	
RP401	1030	-		
RP403	1043	-	1037	
RP410	1064	-	1046	
RP425	1103	-	1112	
RP430	1104	-	1161	
RP440	1152	-	1140	
RP450	1155	-	1116	1114
RP460	1157	-	1154	
RP470	1162	-	1130	
RP490	1176	-	1173	
RP490	1204	-	1175	
RP500	1225	-	1222	
RP550	1262	-	1235	
RP560	1267	-	1263	
RSTKCA	1320	-		
RSTS35	1644	-	1633	
RTKC10	1330	-	1365	
RTKC15	1340	-	1367	
RTKC20	1362	-	1355	
RTKC30	1370	-	1347	1337
RTKC40	1404	-	1371	
RTKC45	1406	-	1475	1417
RUSR25	1461	-	1450	
RUSR30	1471	-	1446	
RUSR40	1473	-	1470	1460 1441
RWCHK	225	-		
RWCK20	232	-		
SKPDEV	1667	-	1677	
SKPF10	344	-	350	
SKPF20	351	-	347	
SKPF30	352	-	345	
SKPFRM	341	-	361	
STS10	1731	-	1745	
STS20	1735	-	1733	
UPNFER	420	-	424	
UPROM	432	-	426	
URTP	402	-		
URTP00	403	-	375	
URTP10	430	-	414	
URTP15	435	-	431	
URTP20	437	-	427	
URTP30	534	-	531	
URTP40	537	-	535	
URTP70	554	-	570	
URTP80	573	-	556	
URTPV	374	-		
URTS	1502	-		

ENTRY TABLE

CKSMER	1246	-
CKSUMS	1242	-
CLM10	1277	-
CLNMEM	1271	-
CLYAIL	1310	-
CX-AX	362	-
FLSCH	13	-
FLSCH0	12	-
FLSCH2	23	-
FLSCH3	10	-
FLSCH1	2	-
FLSCH4	5	-
FLSCH8	16	-
FLSHDT	203	-
FLSHRT	173	-
FLTYER	212	-
ANTDEV	1657	-
INSTAT	1710	-
NORMCK	705	-
NOROOM	716	-
RDENT	241	-
READP	620	-
READS	1565	-
READSB	611	-
REN110	243	-
RG-SY#	1761	-
RP335	1002	-
RP400	1026	-
RP401	1030	-
RSTKCA	1320	-
RUCHK	225	-
SKPFRM	341	-
UPROM	432	-
URTP	402	-
URTPV	374	-
URTS	1502	-

EXTERNAL REFERENCES

ANNOUT	1753				
ANNOUT	1754				
AOUT1	407				
AOUT1	410				
AOUTFL	35				
AOUTFL	36				
AOUTIN	14	443			
AOUTIN	15	444			
ASP	1663				
ASP	1664				
ASRCH	415				
ASRCH	416				
CKSMER	1242				
CKSMER	1243				
CKSUME	1615				
CKSUME	1616				
CLM10	1024				
CLM10	1025				
CLNMEM	1227				
CLNMEM	1230				
CLTAIL	1010	1165			
CLTAIL	1011	1166			
CNTBYT	511	700			
CNTBYT	512	701			
COPYBF	171				
COPYBF	172				
CPCM10	661				
CPCM10	662				
CPCMHO	470				
CPCMHO	471				
CRTFL	537				
CRTFL	540				
CRTFLO	1520				
CRTFLO	1521				
CSECKUT	600				
CSECKUT	601				
CSECKX	1240				
CSECKX	1241				
CSECE0	1244				
CSECE0	1245				
CSECEX	143	727			
CSECEX	144	730			
CSNCFD	0				
CSNCFD	1				
CSRDY	32				
CSRDY	33				
CX-AX	256				
CX-AX	257				
DDT0	241				
DDT0	242				
DECADA	1056	1060	1152	1163	1375 1377
DECADA	1057	1061	1153	1164	1376 1400
DSFEFP	223	1642			
DSFEER	224	1643			
DUPFL	234				
DUPFL	235				

ENCP00	1320	1412		
ENCP00	1321	1413		
ERROR	432			
ERROR	433			
ERRPR	463			
ERRPR	464			
FDEV20	1671			
FDEV20	1672			
FLINKA	460			
FLINKA	461			
FLINKP	643			
FLINKP	644			
FLNMR	420			
FLNMR	421			
FLSCH	1503			
FLSCH	1504			
FLSCH0	623	1567		
FLSCH0	624	1570		
FLSCHX	446			
FLSCHX	447			
FNDCA8	16			
FNDCA8	17			
FNDEN0	1526			
FNDEN0	1527			
FNID60	1755			
FNID60	1756			
FNTDEV	115			
FNTDEV	116			
GCPK00	1466			
GCPK00	1467			
GENLNK	1133	1206		
GENLNK	1134	1207		
GETPC	435			
GETPC	436			
GTBYTA	1047	1123		
GTBYTA	1050	1124		
GTFEND	627	657	1404	
GTFEND	630	660	1405	
GTLINK	1407			
GTLINK	1410			
INCAD2	506	1121	1141	1425
INCAD2	507	1122	1142	1426
INCADA	647	760	1143	1310
INCADA	650	761	1144	1311
LDSET0	472	707		
LDSET0	473	710		
LUJLY	1260			
LUJLY	1261			
MEMLFT	664			
MEMLFT	665			
MESSLP	1246			
MESSLP	1247			
MSGROM	434			
NATNRO	243			
NATNRO	244			
NFRPU	1267			
NFRPU	1270			
NRD	337	766		
NRD	340	767		
NRDC	1606			

NRDC	1607				
NXBYTA	560	651	1105	1427	
NXBYTA	561	652	1106	1430	
NXLDEL	1157				
NXLDEL	1160				
PACKE	714				
PACKE	715				
PILEER	571				
PILEER	572				
PLERCK	251				
PLERCK	252				
PLERCK	125	212	716	1634	
PLERCK	126	213	717	1635	
PTBYTA	762	1054	1146	1313	1454
PTBYTA	763	1055	1147	1314	1455
PURGEF	237				
PURGEF	240				
PUTPEX	1041				
PUTPEX	1042				
R5-R6	21				
R5-R6	22				
RDDFFM	246	330	746		
RDDFFM	247	331	747		
RDLPBK	50				
RDLPBK	51				
RDREG	1600				
RDREG	1601				
RDRGA	1602				
RDRGA	1603				
RDTYPE	1716				
RDTYPE	1717				
RENT10	54				
RENT10	55				
RENTFH	101				
RENTFH	102				
RECAOR	162				
RECAOR	163				
RG-BY#	1514				
RG-BY#	1515				
RSTKCA	1232				
RSTKCA	1233				
RTNP+2	1700				
RTNF+2	1701				
RWCHK	452	1507			
RWCHK	453	1510			
SCHDEV	1710				
SCHDEV	1711				
SDATA	565	576			
SDATA	566	577			
SEEKRE	76				
SEEKRE	77				
SEEKRD	167				
SEEKRD	170				
SEEKRN	731	1571			
SEEKRN	732	1572			
SEKSGU	541	1522			
SEKSGU	542	1523			
SETBPG	46				
SETBPG	47				
SIZSGU	1630				

```

SIZESUB 1631
SKPFM 265 274 312 314 317 321 324 326
SKPFM 266 275 313 315 320 322 325 327
SHDATA 733
SHDATA 734
SREMAP 1361 1471
SREMAP 1362 1472
TALKER 1712
TALKER 1713
TBITMA 1357 1443
TBITMA 1360 1444
UNT 705 1026 1610
UNT 706 1027 1611
UNCHK 1725
UNCHK 1726
UFLINK 1414
UFLINK 1415
WKUP90 1265
WKUP90 1266
WRTA20 1556
WRTA20 1557

```

End of VASM assembly

```

*****
*****
*****

```

VASM ROM ASSEMBLY REV. 6/81A

OPTIONS: L C S

2 FILE SCPL4B

```

*
*
* *****
* FINDID - GIVEN A DEVICE ID IN ALPHA REG, RETURN WITH THE DEVICE *
* LOOP ADDRESS IN X-REG. *
* X-REG = 0, IF THE DEVICE ID NOT FOUND *
* THE SEARCH ALWAYS START FROM DEVICE #1 *
* *****
*
12 ENTRY FINDID
13 ENTRY FNID10 ** ADD ON JUNE 4, 1981
*
15 0 204 CON 0204 D
16 1 11 CON 011 I
17 2 4 CON 004 D
18 3 16 CON 016 N
19 4 11 CON 011 I
20 5 6 CON 006 F
21 6 FINDID 1 GOSUB SCHDEV
21 7 0
22 10 10 S3= 1
23 11 1 GOSUB AOUT1 GET THE ID FROM A-REG
23 12 0
24 13 FNID10 1 GOSUB TALKER
24 14 0
25 15 1 GOSUB PLERCK
25 16 0
26 17 144 HPL=CH 1

```

```

27 20      1205 CH=      0241
28 21      244 HPL=CH 2
29 22      611 CH=      0142      SEND "SID"
30 23      404 SS=      0
31 24      116 C=0
32 25      160 N=C
33 26      1004 S2=      0
34 27      1 GOSUB      INADRD      READ THE DEVICE ID
34 30      0
35 31      1434 PT=      1
36 32      260 C=N
37 33      1356 ? C#0    W
38 34      113 GONC      FNID35 ( 45)
39 35 FNID20 1352 ? C#0    WPT      RIGHT JUSTIFY THE ID IN N
40 36      37 GOC        FNID30 ( 41)
41 37      1074 RCR      2
42 40      1753 GOTO     FNID20 ( 35)
43 41 FNID30 416 A=C      W
44 42      630 C=N
45 43      1556 ? A#C    W      FOUND THE ID ?
46 44      73 GONC      FNID45 ( 53) YES
47 45 FNID35 1 GOSUB     PILEN
47 46      0
48 47      1 GOSUB      NXTDEV      GET NEXT DEVICE ADDRESS
48 50      0
49 51      63 GOTO      FNID40 ( 57) NO NEXT DEVICE
50 52      1413 GOTO     FNID10 ( 13) SEARCH NEXT DEVICE
51 53 FNID45 544 C=HPIL 5      GET DEVICE ADDRESS
51 54      572
51 55      503
52 56      53 GOTO      FNID65 ( 63)

*
54 57 FNID40 56 B=0      W
55 60 FNIDRT 1 GOLONG    RCL
55 61      2

*
57      ENTRY FNID60

*
59 62 FNID60 316 C=B      GET DEVICE ADDR
60 63 FNID65 1 GOSUB     BINBD0      NORMALIZE THE BINARY
60 64      0
61 65      36 A=0        S
62 66      334 PT=       10
63 67      12 A=0        WPT
64 70      106 C=0        X
65 71      1160 DADD=C
66 72      1046 C=C+1     X      ASSUME GREATER THAN 9
67 73      406 A=C        X
68 74      1534 PT=       12
69 75      1502 ? A#0     PT
70 76      37 GOC        FNID70 ( 101)
71 77      1772 A SL      M
72 100      6 A=0        X
73 101 FNID70 156 AB EX   W
74 102      1563 GOTO     FNIDRT ( 60)

```

```

*
*****
* STDFIO - SEND IFC (INTERFACE CLEAR)
*****
*

```

80 ENTRY STOPIO

```

83 103      217 CON      0217      0
84 104      11 CON      011        I
85 105      20 CON      020        P
86 106      17 CON      017        O
87 107      24 CON      024        T
88 110      23 CON      023        S
89 111 STOPIO      1 GOSUB  SCHDEV
89 112      0

```

91 ENTRY IFC

```

7C4B 93 113 IFC      44 HPL=CH 0
94 114      1601 CH=    0340 * E0
95 115      144 HPL=CH 1
96 116      1005 CH=    0201      SET UP FOR CMD FRAME
97 117      244 HPL=CH 2
98 120      1101 CH=    0220      SEND CMD - IFC
99 121      1034 PT=    2
100 122      1520 LC     13      TIME OUT ONLY 1 SECOND
101 123      1 GOSUB  SCMD20
101 124      0
102 125      44 HPL=CH 0      SET CLIFCR=1
103 126      1611 CH=    0342
104 127      1740 RTN

```

7C4J

 * PURGEF - PURGE A FILE (NON-PROGRAMMABLE) *
 * A FILE CAN'T BE PURGED IF IT IS PROTECTED *

```

111      ENTRY  PURGEF
112      ENTRY  PURGEF
113      ENTRY  REWENT
114      ENTRY  WRET10
115      ENTRY  WRET15

```

```

117 130      205 CON      0205      E
118 131      7 CON      007        G
119 132      22 CON      022        R
120 133      25 CON      025        U
121 134      20 CON      020        P
122 135 PURGEF      1 GOSUB  FLSCHJ      SEARCH THE FILE
122 136      0
123 137      76 B=0      S      DO COPYBF
124 140 PURGEF      1 GOSUB  CHKPCT      CHECK IF THE FILE PROTECTED
124 141      0
125 142      260 C=N
126 143      136 C=0      S
127 144      160 N=C

```

*
 * REWENT - WRITE A FILE ENTRY TO DIRECTORY
 * ASSUME : THE POINTER IS POINTING AT END OF A FILE ENTRY.
 * THIS ROUTINE WILL WRITE THE ENTRY OVER THE SAME PLACE.
 * INPUT : N = FILE NAME
 * N = FILE TYPE(2), STARTING REC.#(4), FILE LENGTH IN RECS(4)
 * FILE SIZE(4)
 * IF B.S = 0, WILL DO A COPY BUFFER A TO BUFFER B AT END

* USED A, B,X, C, PT, S0-S7 +2 SUB LEVEL

*

765138	145	REWENT	1	GOSUB	REQADR	READ CURRENT ADDR
139	146		0			
139	147		646	A=A-1	X	BACK UP 1 RECORD
140	150		1312	? B#0	WPT	POINTING AT REC. BOUNDARY ?
141	151		77	GOC	WRET10 (160)	NO
142	152		646	A=A-1	X	
143	153		206	B=A	X	
144	154		1	GOSUB	SEEKRD	BACK UP 2 REC. & READ IT
144	155		0			
145	156		146	AB EX	X	
146	157		46	B=0	X	
147	160	WRET10	1	GOSUB	SEEK	
147	161		0			
148	162		1	GOSUB	SRWRT	
148	163		0			
149	164		146	AB EX	X	AC1:01 = PTR OF AN ENTRY
150	165		460	LDI		
151	166		40	CON	32	
152	167		706	A=A-C	X	BACK 32 BYTES
153	170		0	NOP		
154	171	WRET15	1	GOSUB	SETBPT	SET BYTE POINTER
154	172		0			
155	173		1	GOSUB	DTFLOW	
155	174		0			
156	175		630	C=M		
157	176		534	PT=	6	
158	177		1	GOSUB	SNBYTS 7FBC	SEND 7 BYTES OF NAME
158	200		0			
159	201		260	C=N		
160	202		1076	C=C+1	S	IS IT A 41C FILE ?
161	203		717	GOC	WRET50 (274)	NO, DON'T CHANGE TYPE
162	204		260	C=N		C.S = FILE TYPE
163	205		416	A=C		
164	206		132	C=0	M	
165	207		106	C=0	X	
166	210		474	ROR	8	C= 00000000T00000
167	211		1334	PT=	13	
168	212		220	LC	2	LOAD 3 BLANK (HEX 20)
169	213		20	LC	0	
170	214		220	LC	2	
171	215		20	LC	0	
172	216		220	LC	2	C= 20202000T00000
173	217		676	A=A-1	S	IS A PURGED FILE ?
174	220		117	GOC	WRET40 (231)	YES
175	221		676	A=A-1	S	IS IT THE ASCII FILE ?
176	222		53	GONC	WRET30 (227)	
177	223		234	PT=	5	
178	224		20	LC	0	
179	225		120	LC	1	C= 20202000010000
180	226		33	GOTO	WRET40 (231)	
181	227	WRET30	1234	PT=	7	
182	230		1620	LC	14	C= 202020E0T00000
183	231	WRET40	134	PT=	4	WRITE 5 BYTES
184	232		1	GOSUB	SNBYTS 7FBC	SEND 2 BYTES OF TYPE
184	233		0			
185	234		260	C=N		
186	235		1574	ROR	12	
187	236		34	PT=	3	

188	237	112	C=0	WPT	
189	240	174	ROR	4	
190	241	1	GOSUB	SNBYTES	SEND 4 BYTES OF START REC.#
190	242	0			
191	243	260	C=N		
192	244	474	ROR	8	CI[13:10]=FILE LENGTH IN REC.S
193	245	34	PT=	3	
194	246	112	C=0	WPT	
195	247	174	ROR	4	
196	250	1	GOSUB	SNBYTES	SEND 4 BYTES OF REC. LENGTH
196	251	0			
197	252	116	C=0	W	
198	253	234	PT=	5	
199	254	1	GOSUB	SNBYTES	SEND 6 BYTES OF ZEROS(TIME)
199	255	0			
200	256	116	C=0	W	
201	257	1334	PT=	13	
202	260	1020	LC	8	
203	261	334	PT=	10	
204	262	120	LC	1	
205	263	1434	PT=	1	
206	264	1	GOSUB	SNBYTES	SEND 2 BYTES OF VOLUME NUMBER
206	265	0			
207	266	260	C=N		
208	267	136	C=0	S	ZERO FILE TYPE
209	270	174	ROR	4	
210	271	1034	PT=	2	SEND 2 BYTES OF SIZE AND
211	272	1	GOSUB	SNBYTES	1 BYTE OF FILE OPTION
211	273	0			
212	274	144	HPL=CH	1	
213	275	405	CH=	@101	MAKE LAST BYTE AN END FRAME
214	276	460	LDI		
215	277	40	CON	@40	
216	300	1	GOSUB	SDATA	
216	301	0			
217	302	1	GOSUB	WAITS	WAIT FOR DONE AND CHECK ERROR
217	303	0			
218	304	1336	? B#0	S	UPDATE DIR BUFR ?
219	305	1540	RTN C		NO, IF B.S # 0

本
學
學
學
學
學
學
學
學
學
學
學
學
學

COPYBF - COPY THE CURRENT DATA BUFFER TO DIRECTORY BUFFER
FILBER HAS TO 256 BYTES BUFFER, ONE IS USED FOR TRANSFERRING
DATA TO/FROM TAPE, ONE IS USED BY READ/WRITE LOOP BACK
FUNCTION WHICH IS KIND OF NOT USED BY FILBER. THE 2ND BUFR
IS NOW USED TO KEEP LAST ACCESSED DIRECTORY RECORD WHICH
CAN SPEED UP THE FILE SEARCH, IF THE # OF FILES ON THE
TAPE IS SMALL

USED A, C, +2 SUB LEVELS
OUTPUT : STORE A CONSTANT 250 TO BYTE #250 TO INDICATE LAST FILE
 ACCESSED WAS NOT A DATA FILE
 LEAVE THE CST NOT A TALKER NOR A LISTENER

235		ENTRY	COPYBF	
237	306	COPYBF	1	GOSUB REQADR READ CURRENT ADDR
237	307		0	
238	310		1	GOSUB LISTEN
238	311		0	

239	312	460	LDI		
240	313	251	CON	@251	SEND DDL 09 - COPY BUF1 TO BUF2
241	314	1	GOSUB	SCMD	
241	315	0			
242	316	460	LDI		
243	317	372	CON	250	
244	320	1	GOSUB	SETBPC	BYTE POINTER = 252
244	321	0			
245	322	1	GOSUB	WRLPBK	GOTO WRITE LOOP BACK MODE
245	323	0			
246	324	460	LDI		
247	325	372	CON	250	
248	326	1	GOSUB	SDATA0	
248	327	0			
249	330	146	AB EX	X	
250	331	1	GOSUB	SETBPL	SET THE BYTE PTR BACK
250	332	0			
251	333	1	GOLONG	UNL	
251	334	2			

*
 * CHKCST - LOOK FOR THE CASSETTE IN THE LOOP AND THEN READ ITS STATUS
 * TO SEE IF IT IS IN IDLE.
 * IF CASSETTE NOT FOUND OR CASSETTE IS BUSY WILL DIRECT DO
 * AN ERROR EXIT TO COCONUT MAINFRAME NOT RETURNING TO
 * CALLING PROGRAM.

* ASSUME NOTHING
 * OUTPUT : CASSETTE STATUS IN S0-7
 * CASSETTE ADDR IN R5
 * USED A,C,S0-7 NO PT, +2 SUB LEVEL

267	ENTRY	CHKCST	CHECK CASSETTE PRESENT AND READY
264	ENTRY	CSNOFD	CASSETTE NOT FOUND
265	ENTRY	CSEREX	CASSETTE ERROR EXIT
266	ENTRY	CSERE0	
267	ENTRY	CSRDY	CHECK IF CASSETTE READY
268	ENTRY	INTDIR	
269	ENTRY	CHKCS0	

271	335	CHKCST	404	S8=	0	
272	336	CHKCS0	1	GOSUB	FNDCCAS	LOOK FOR THE CASSETTE
272	337		0			
273	340		243	GOTO	CSNOFD (364)	CASSETTE NOT FOUND
274	341	CSRDY	214	?S8=1		CASSETTE BUSY ?
275	342		63	GONC	CSRDY1 (350)	NO
276	343		1	GOSUB	CSSTAS	
276	344		0			
277	345		1	GOSUB	PLERCK	
277	346		0			
278	347		1723	GOTO	CSRDY (341)	
279	350	CSRDY1	414	?S8=1		
280	351		1540	RTN C		
281	352		460	LDI		SEE IF NEW TAPE
281	353		7	CON	7	
283	354		406	A=C	X	
284	355		1634	PT=	0	
285	356		1630	C=ST		
286	357		1542	? A#C	PT	
287	360		1540	RTN C		
288	361	INTDIR	1	GOSUB	SEEKR2	RESTORE DIR BUFR FROM REC. 0
288	362		0			

```

289 363          1233 GOTO    COPYBF ( 306 )
290 364 CSNOFD    1 GOSUB    PLEREX          SAY "NO CASSETTE"
290 365          0
291 366          16 CON      @16            N
292 367          17 CON      @17            O
293 370          40 CON      @40
294 371          4 CON      @04            D
295 372          22 CON      @22            R
296 373          11 CON      @11            I
297 374          26 CON      @26            V
298 375          1005 CON     @1005          E
299 376 CSEREX    1 GOSUB    LEFTJ          LEFT JUSTIFY DISPLAY
299 377          0
300 400 CSERE0    410 SS=      1
301 401          1 GOSUB    MSG105          PRINT ERROR MESSAGE
301 402          0
302 403          1 GOLONG   ERR110
302 404          2

```

```

*
304          ENTRY    DSPERR

```

```

306 405 DSPERR    1 GOSUB    MESSL
306 406          0
307 407          40 CON      @40
308 410          5 CON      @05            E
309 411          22 CON      @22            R
310 412          1022 CON     @1022          R
311 413          1633 GOTO    CSEREX ( 376 )

```

```

*
* CHKPC - CHECK IF THE FILE IS PROTECTED
* IN EVERY FILE ENTRY THERE ARE TWO BITS IN THE FILE TYPE IS USED
* FOR FILE PROTECTION. THE MSB (BIT 7) OF THE FILE TYPE IS THE USER
* PROTECT BIT WHICH CAN BE SET OR RESET BY THE USER THRU THE TWO
* PROTECT FUNCTIONS.

```

```

319          ENTRY    CHKPC          CHECK IS FILE PROTECTED
*
321 414 CHKPC     260 C=N          CHECK IF THE FILE IS PROTECTED
322 415          1374 RCR      13
323 416          776 C=C+C      S
324 417          1640 RTN NC          NO
325 420 FLPT10    1 GOSUB    PLEREX
325 421          0
326 422          6 CON      @06            F
327 423          14 CON      @14            L
328 424          40 CON      @40
329 425          23 CON      @23            S
330 426          5 CON      @05            E
331 427          3 CON      @03            C
332 430          25 CON      @25            U
333 431          22 CON      @22            R
334 432          5 CON      @05            E
335 433          1004 CON     @1004          D
336 434          1423 GOTO    CSEREX ( 376 )

```

```

*
*****
* VERIFY - VERIFY A FILE
* WILL ONLY READ THE FILE INTO DRIVER'S BUFFER NOT TRANSMIT
* THE DATA OUT. BECAUSE ANY PIL TRANSMIT ERROR SHOULD BE
* DETECTED WHILE GENERATING THE FILE, SO THE ONLY THING

```

MIGHT WENT WRONG WAS THE DRIVER DID NOT RECORD ON THE TAPE*
CORRECTLY.

```

347          ENTRY  VERIFY
348 435      231 CON   @231      Y
349 436        6 CON   @06      F
350 437       11 CON   @11      I
351 440       22 CON   @22      R
352 441        5 CON   @05      E
353 442       26 CON   @26      V

355 443 VERIFY    1 GOSUB  FLSCHI      READ THE FILE ENTRY
356 444          0
357 445          1 GOSUB  SEEKRN      SEEK & READ 1ST RECORD
358 446          0
359 447       260 C=N
360 450       174 RCR      4      C.X= # OF RECS IN FILE
361 451      1146 C=C-1    X      B.X= # OF RECS -1
362 452       346 BC EX   X
363 453 VERF10    146 AB EX  X
364 454       646 A=A-1   X      ALL DONE ?
365 455          1 GOLF   UNT      YES
366 456          3
367 457       206 B=A     X
368 460          1 GOSUB  SEEK40      READ ONE RECORD
369 461          0
370 462      1713 GOTO   VERF10 ( 453)

```

WRTA - WRITE ALL

```

372          ENTRY  WRTA
373          ENTRY  WRTA20

375 463      201 CON   @201      A
376 464       24 CON   @24      T
377 465       22 CON   @22      R
378 466       27 CON   @27      W
379 467 WRTA    314 ?S10=1      IN ROM ?
380 470          1 GOLF   WPRON    NOT ALLOW WHILE POINTING ROM
381 471          3
382 472      1770 C=REGN 15
383 473       106 C=0     X
384 474      1146 C=C-1   X      DESTROY LINE NUMBER
385 475      1750 REGN=C 15
386 476          1 GOSUB  GTFEND    GET FINAL END
387 477          0
388 500 WRTA10 → 1 GOSUB  FLINKA
389 501          0
390 502      1514 ?S12=1      ANY PRIVATE PRGM ?
391 503          1 GOLF   ERRPR     YES
392 504          3
393 505      1506 ? A#0     X      REACH REG.0 ?
394 506      1727 GOC      WRTA10 ( 500) NOT YET
395 507       116 C=0
396 510      1160 DADD=C
397 511          1 GOSUB  FLSCH     CHECK DUPLICATION
398 512          0

```

394	513	460	LDI			
395	514	4	CON	4		W/ALL FILE TYPE
396	515	1	GOSUB	RWCHK		CHECK FOR OVER WRITE
396	516	0				
397	517	1	GOSUB	FNDEND		GET ADDR OF TOP OF MEM
397	520	0				
398	521	116	C=0	W		
399	522	1160	DADD=C			
400	523	1670	C=REGN	14		
401	524	674	ROR	11		
402	525	1530	ST=C			S0= USER FLAG 11
403	526	116	C=0	W (ALL)		
404	527	460	LDI			
405	530	257	CON2	10	15	04F
406	531	1106	C=A-C	X		C,X = TOTAL # OF REGS
407			LEGAL			
408	532	1	GOSUB	RG-BY#		C[5 2] # OF REGS PT-4
408	533	0				
409	534	1074	ROR	2		C[3:0]
410	535	1152	C=C-1	WPT RE		# OF REGS - 1
411	536	1574	ROR	12		C[5 2] # OF REGS - 1
412	537	1614	S0=1			WANT TO SET UP AUTO RUN ?
413	540	33	GONC	WRTA15 (543)		NO
414	541	1634	PT=	0		
415	542	320	LC	2		
416	543	1434	PT=	1		
417	544	420	LC	4		LOAD FILE TYPE
418	545	1	GOSUB	CRTFL		CREAT THE FILE
418	546	0				
419	547	1	GOSUB	SEKSUB		SEEK TO BEGINNING OF FILE
419	550	0				
420	551	116	C=0			SET UP TO WRITE CHIP 0 — 7D60
421	552	460	LDI			
422	553	20	CON	16	010	
423	554	530	M=C			
424	555	56	B=0			INITIALIZE CHECKSUM
425	556	410	S8=	1		
426	557	1	GOSUB	SNDRCGA		SEND CHIP 0
426	560	0				
427	561	260	C=N			C[3:0]= TOTAL # OF REGS
428	562	74	ROR	3		
429	563	460	LDI			
430	564	300	CON2	12	0	0C0
431	565	1574	ROR	12		
432	566	1146	C=C-1	X		SUBSTRACT 16 REGS
433	567	1374	ROR	13		
434	570	530	M=C			
435	571	410	S8=	1		
436	572	1	GOSUB	SNDRCGA		
436	573	0				
437	574	316	C=B			
438	575	1	GOSUB	SNDRCG		WRITE CHECKSUM
438	576	0				
439	577	1	GOLONG	SNDRDH		CLOSE FILE
439	600	2				

* READA - READ ALL *

*

445		ENTRY	READA		
446		ENTRY	COLDER		
448	601	201 CON	@201		A
449	602	4 CON	@04		D
450	603	1 CON	@01		A
451	604	5 CON	@05		E
452	605	22 CON	@22		R
453	606	READA 460 LDI			
454	607	4 CON	4		WRITE ALL FILE TYPE
455	610	1 GOSUB	FLSCH0		SEARCH THE WRITE-ALL FILE
455	611	0			
456	612	1 GOSUB	FNDEND		FIND ADDR OF TOP MEM
456	613	0			
457	614	460 LDI			
458	615	260 CON2	11 0		04C
459	616	706 A=A-C	X		A.X=TOTAL # OF REGS IN 410
460	617	260 C=N			GET FILE SIZE
461	620	1406 ? A<C	X		ENOUGH ROOM TO READ IT IN ?
462	621	1 GOLC	NOROOM		NO, SAY "NO ROOM"
462	622	3			
463	623	1 GOSUB	SEEKRN		SEEK TO THE FILE
463	624	0			
464	625	116 C=0			
465	626	460 LDI			
466	627	20 CON	16		READ 16 REGS OF CHIP 0
467	630	530 M=C			
468	631	56 B=0	W		INITIALIZE CHECKSUM
469	632	1 GOSUB	RDREG 7610		
469	633	0			
470	634	260 C=N			C[3:0]= FILE SIZE
471	635	74 RCR	3		
472	636	460 LDI			
473	637	300 CON2	12 0		
474	640	1574 RCR	12		
475	641	1146 C=C-1	X		SUBSTRACT 16 REGS
476	642	1374 RCR	13		
477	643	1 GOSUB	RDRG10 7613		
477	644	0			
478	645	1 GOSUB	RDRGA 763E		READ CHECKSUM
478	646	0			
479	647	160 N=C			SAVE CHECKSUM IN N
480	650	674 RCR	11		
481	651	1 GOSUB	NRDC		SEND "NRD"
481	652	0			
482	653	1 GOSUB	UNT		
482	654	0			
483	655	260 C=N			
484	656	156 AB EX	W		
485	657	1556 ? A#C	W		CHECKSUM MATCH ?
486	660	53 GONC	RALL10 (665)		YES
487	661	1 GOSUB	CKSMER		SAY "CHECKSUM ERR"
487	662	0			
488	663	1 GOLONG	COLDST		DO COLD START
488	664	2			
489	665	RALL10 1504	S12= 0		
490	666	304	S10= 0		
491	667	1 GOSUB	LDSST0		GET USER FLAG 11
491	670	0			
492	671	674 RCR	11		

RAILS

7DAD

COLDER

493	672	1530	ST=C	S0= USER FLAG 11
494	673	1614	?S0=1	WAS IT SET ?
495	674	1	GOLC WKUP30	YES, SET OFF TO RUN
495	675	3		
496	676	1	GOLONG NFRC	
496	677	2		

*

* WRTK - WRITE KEYS

*

502		ENTRY	WRTK	
504	700	213 CON	0213	K
505	701	24 CON	024	T
506	702	22 CON	022	R
507	703	27 CON	027	W
508	704 WRTK	136 C=0	S	
509	705	1 GOSUB	FLSCH	CHECK FILE DUPLICATION
509	706	0		
510	707	460 LDI		
511	710	5 CON	5	
512	711	1 GOSUB	RWCHK	CHECK FOR OVER WRITE
512	712	0		
513	713	1570 C=REGN	13	
514	714	346 BC EX	X	B.X = CHAIN HEAD
515	715	460 LDI		
516	716	277 CON2	11 15	SEE HOW MANY KEY REGS
517	717 WRTK10	1046 C=C+1	X	
518	720	1160 DADD=C		
519	721	246 AC EX	X	
520	722	1446 ? A<B	X	REACH CHAIN HEAD ?
521	723	63 GONC	WRTK15 (731)	YES
522	724	70 C=DATA		
523	725	246 AC EX	X	
524	726	1076 C=C+1	S	IS THIS A KEY REG ?
525	727	1707 GOC	WRTK10 (717)	YES
526	730	406 A=C	X	
527	731 WRTK15	460 LDI		
528	732	300 CON2	12 0	
529	733	706 A=A-C	X	A.X= # OF KEY REGS
530	734	1506 ? A#0	X	# OF KEY REGS = 0 ?
531	735	243 GONC	WRTK20 (761)	YES
532	736	246 AC EX	X	
533	737	132 C=0	M	
534	740	1 GOSUB	RG-BY#	CONVERT TO # OF BYTES
534	741	0		
535	742	1434 PT=	1	
536	743	520 LC	5	LOAD FILE TYPE
537	744	1 GOSUB	CRTFLO	CREAT THE KEY FILE
537	745	0		
538	746	1 GOSUB	SEKSUB	SEEK TO THE FILE
538	747	0		
539	750	260 C=N		C.X = # OF KEY REGS
540	751	74 RCR	3	
541	752	460 LDI		
542	753	300 CON2	12 0	START WRITING REG ADDR
543	754	674 RCR	11	
544	755	530 M=C		
545	756	56 B=0	W	INITIALIZE CHECKSUM

```

546 757      1 GOLONG WRTA20
546 760      2
547 761 WRTK20 1 GOSUB PLEREX      SAY "NO KEYS"
547 762      0
548 763      16 CON      @16      N
549 764      17 CON      @17      O
550 765      40 CON      @40
551 766      13 CON      @13      K
552 767      5 CON      @05      E
553 770      31 CON      @31      Y
554 771      1023 CON     @1023     S
555 772      1 GOLONG CSEREX
555 773      2

```

*

* READK - READ IN A "KEY" FILE

*

```

561      ENTRY  READK

563 774      213 CON      @213      K
564 775      4 CON      @04      D
565 776      1 CON      @01      A
566 777      5 CON      @05      E
567 1000     22 CON      @22      R
568 1001 READK 460 LDI
569 1002      5 CON      5
570 1003      1 GOSUB     FLSCHO     SEARCH THE "KEY" FILE
570 1004      0
571 1005      1 GOSUB     SEEKRN     SEEK AND READ THE RECORD
571 1006      0
572 1007     1570 C=REGN 13      GET CHAIN HEAD
573 1010     346 BC EX  X      SAVE IT IN B.X
574 1011      1 GOSUB     MENLFT     COMPUTE # OF ZERO REGS
574 1012      0
575 1013     146 AB EX  X
576 1014     706 A=A-C  X      A.X= ADDR OF LAST ZERO REG
577 1015     260 C=N
578 1016     132 C=0      M
579 1017     674 RCR      11
580 1020     272 AC EX  M
581 1021     260 C=N
582 1022     174 RCR      4      SAVE FILE SIZE IN N[13:10]
583 1023     460 LDI
584 1024     300 CON2     12      0
585 1025 RDKY10 1160 DADD=C     ENABLE NEXT KEY REG
586 1026     160 N=C
587 1027     70 C=DATA
588 1030     1076 C=C+1  S      IS THIS A KEY REG ?
589 1031     133 GONC     RDKY40 (1044) NO, MAY BE DONE OR LIFT IT UP
590 1032 RDKY15 1532 ? A#0  M      DONE WITH REGS IN FILE
591 1033     53 GONC     RDKY30 (1040) YES, REPLACE BY A BLANK KEY REG
592 1034     672 A=A-1  M      NO, DEC. # OF REGS IN FILE
593 1035 RDKY20 260 C=N      GET ADDR OF CURRENT KEY REG
594 1036     1046 C=C+1  X      POINT TO NEXT KEY REG
595      LEGAL
596 1037     1663 GOTO     RDKY10 (1025)

```

* EXISTING # OF KEY REGS > REGS IN FILE, REPLACE A BLANK KEY REG

* IN THOSE EXTRA OLD KEY REGS

```

598 1040 RDKY30 116 C=0      W

```



```

600 1041          1176 C=C-1  S
601 1042          1360 DATA=C
602 1043          1723 GOTO   RDKY20 (1035) LOOK AT NEXT KEY REG
603 1044 RDKY40 1532 ? A#0  M          DONE WITH REGS IN FILE ?
604 1045          163 GONC   RDKY50 (1063) YES, READY TO READ
* EXISTING KEY REGS < REGS IN FILE, PRECREAT BLANK KEY REG BY
* LIFT UP ONE REG FOR EACH ADDITIONAL KEY REG
607 1046          116 C=0    W
608 1047          1160 DADD=C
609 1050          1176 C=C-1  S
610 1051          356 BC EX   W
611 1052          1570 C=REGN 13          GET CHAIN HEAD
612 1053          1546 ? A#C  X          LAST ZERO REG= CHAIN HEAD ?
613 1054          1 GOLNC   NORMCK       YES, SAY "NO ROOM" OR "TRY AGAIN"
613 1055          2
614 1056          260 C=N
615 1057          1 GOSUB   ASN15        LIFT UP ONE REG
615 1060          0
616 1061          546 A=A+1  X          INC. ADDR OF LAST ZERO REG
617          LEGAL
618 1062          1503 GOTO   RDKY15 (1032) DEC. REGS & GOTO NEXT KEY REG
*
620 1063 RDKY50 260 C=N          SET M.X= # OF REGS TO READ
621 1064          374 RCR      10          M.M= STRAT LOADING ADDR
622 1065          132 C=0    M
623 1066          134 PT=    4
624 1067          1420 LC     12
625 1070          530 M=C
626 1071          1 GOSUB   RDREG0
626 1072          0
627 1073          1010 S2=    1          RECONSTRUCT ALL KET ASSIGNMENT
628 1074          1 GOSUB   RSTKCA
628 1075          0
629 1076 RDKYER  1 GOLONG  PLERCK
629 1077          2
*
*
* REQADR - READ CURRENT TAPE ADDRESS
* WITH THE CURRENT ADDRESS AND THE STARTING RECORD # OF THE
* FILE, WE CAN COMPUTE WHERE IS FILE POINTER NOW. SO WE CAN
* TELL HOW MANY REGISTER LEFT FROM THE FILE POINTER TO THE
* END OF FILE.
* 1. ADDRESS CASSETTE AS A TALKER
* 2. SEND SEC.CMD- "SEND ADDRESS"
* 3. READ 2 BYTES OF RECORD # AND 1 BYTE OF BYTE #
* 4. COMPUTE # OF REGISTER LEFT IN THE FILE
* INPUT:
* NI(3:0) = CURRENT FILE SIZE IN # OF REGISTERS
* OUTPUT : NI(3:0) = # OF REGISTERS LEFT IN THE FILE
* USED A,B,C +1 SUB LEVEL
*
646          ENTRY  REQADR          GOSUB TALKER
*
648 1100 REQADR  1 GOSUB  TALKER
648 1101          0
649 1102          460 LDI
650 1103          303 CON      @303
651 1104          1 GOSUB   SCMD       SEND SEC.CMD- "SEND ADDRESS"
651 1105          0
652 1106          234 PT=    5

```

```

653 1107      1 GOSUB NATNRD      SEND "NAT"
653 1110      0
654 1111 REQA10 1 GOSUB RDDFRM      READ 1ST BYTE OF RECORD #
654 1112      0
655 1113      1756 A SL
656 1114      1756 A SL
657 1115      266 AC EX XS
658 1116      406 A=C X
659 1117      1724 DEC PT
660 1120      1424 ? PT= 1          READ 4 BYTE YET ?
661 1121      37 GOC REQA15 (1124) YES, THE 4TH BYTE IS ETO
662 1122      1200 HPIL=C 2          ECHO
663 1123      1663 GOTO REQA10 (1111) READ NEXT FRAME
664 1124 REQA15 1104 S9= 0
665 1125      460 LDI
666 1126      100 CON 0100          TEST WAS LAST BYTE ETO
667 1127      1552 ? A#C WPT
668 1130      1467 GOC RDKYER (1076) NO, ERROR
669 1131      1616 A SR
670 1132      1616 A SR
671 1133      212 B=A WPT
672 1134      1616 A SR
673 1135      1616 A SR
674 1136      32 A=0 M
675 1137      1740 RTN

```

```

*
*
*
* DATSUB - FOR FUNCTIONS LIKE WRTR,READR,WRTRX,READRX, THEY USE X-REG
* TO INDICATE STARTING REGISTER AND ENDING REGISTER. THIS
* ROUTINE CHECK THE STARTING & ENDING REGISTER, AND COMPUTE
* # OF REGISTERS NEED TO READ OR WRITE. THERE ARE TWO FATAL
* ERROR MIGHT HAPPEN :
* 1. STARTING OR ENDING REGISTER NOT EXIST - "NONEXISTANT"
* 2. FILE NOT BIG ENOUGH TO CONTENT THE REGS - "NO ROOM"

```

```

* INPUT : X-REG = BBB.EEE WHERE BBB IS THE STARTING REG. # AND EEE
* IS THE ENDING REG. #
* N(11:8)= FILE SIZE IN # OF REGS
* OUTPUT : M.X = # OF REGS TO READ OR WRITE
* M.M = STARTING REGISTER ADDR - 1
* USED A, B,X, C, M, N, +1 SUB LEVEL

```

```

* DATALL - SAME AS DATSUB EXCEPT WILL WRITE ALL THE REGISTERS

```

```

697          ENTRY DATSUB
698          ENTRY DATALL
699          ENTRY DATSER

```

```

*
*
* DFBCK - CHECK WILL THE FOLLOWING READ OR WRITE CROSS THE DATA
* FILE BOUNDARY
* USED A,B,C,N +1 SUB LEVEL
* INPUT : REG.9[5:0]= CURRENT CASSETTE ADDR
* N= FILE ENTRY
* OUTPUT : M.X = # OF REGISTER TO READ OR WRITE
* M[5:3] = STARTING REGISTER ADDRESS
* REG.9[4:0]= # OF BYTES OF CURRENT ADDR PASSED BEGINNING
* FILE

```

		ENTRY	DFBCK	
*	712			
*				
714	1140	DFBCK	1170 C=REGN 9	GET CURRENT ADDR
715	1141		1074 RCR 2	
716	1142		406 A=C X	A.X= CURRENT RECORD #
717	1143		1434 PT= 1	
718	1144		230 C=G	SEE IF LAST TIME A READ OR WRITE
719	1145		1342 ? C#0 PT	LAST OPERATION A WRITE ?
720	1146		27 GOC DFCK10 (1150)	YES
721	1147		646 A=A-1 X	
722	1150	DFCK10	260 C=N	CI[11:8]= SATRTING RECORD #
723	1151		474 RCR 8	C.X= STRATING RECORD #
724	1152		706 A=A-C X	A.X= # OF RECORDS PASSED BOF
725	1153		37 GOC DFCK20 (1156)	NOT POINTING THE DATA FILE
726	1154		374 RCR 10	C.X= # OF RECORDS
727	1155		1406 ? A<C X	CROSSED FILE BOUNDARY ?
728	1156	DFCK20	1 GOLNC FLTYER	YES, ERROR
728	1157		2	
729	1160		1756 A SL	
730	1161		1756 A SL	
731	1162		1170 C=REGN 9	CI[1:0]= BYTE PTR
732	1163		412 A=C WPT	AI[4:0]= # OF BYTES PASSED BOF
733	1164		256 AC EX W	CI[4:0]= # OF BYTES PASSED BOF
734	1165		1150 REGN=C 9	SAVE IN REC.9
*				
736	1166	DATSUB	1 GOSUB FNDEND	FIND 1ST NON-EXIST REG ADDR
736	1167		0	
737	1170		246 AC EX X	
738	1171		530 M=C	SAVE IT IN M.X
739	1172		106 C=0 X	
740	1173		1160 DADD=C	ENABLE CHIP 0
741	1174		370 C=REGN 3	LOAD X-REG
742	1175		1 GOSUB BCDBIN	CONVERT INT(X) TO BINARY
742	1176		0	
743	1177		246 AC EX X	A.X = REG # OF STARTING REG
744	1200		1570 C=REGN 13	GET REG0
745	1201		74 RCR 3	
746	1202		506 A=A+C X	A.X = ABS ADDR OF STARTING REG
747	1203		630 C=M	GET ADDR OF REG LIMIT
748	1204		1406 ? A<C X	1ST REG OVER LIMIT ?
749	1205		57 GOC DATS05 (1212)	NO
750	1206	DATSER	1 GOSUB UNL	
750	1207		0	
751	1210		1 GOLONG ERRNE	
751	1211		2	
752	1212	DATS05	206 B=A - X	B.X=ABS ADDR OF STARTING REG
753	1213		370 C=REGN 3	SHIFT OUT INT. PART OF X-REG
754	1214		416 A=C W	
755	1215		1240 SETDEC	
756	1216		1526 ? A#0 XS	X<1 ?
757	1217		47 GOC DATS20 (1223)	YES
758	1220	DATS10	1772 A SL M	
759	1221		646 A=A-1 X	
760	1222		1763 GONC DATS10 (1220)	
761	1223	DATS20	460 LDI	
762	1224		3 CON 3	C = 1000*FRAC(X)
763	1225		506 A=A+C X	
764	1226		256 AC EX W	
765	1227		1140 SETHEX	

766	1230	1	GOSUB	BCDBIN	CONVERT TO BINARY
766	1231	0			
767	1232	406	A=C	X	A.X = RELATIVE ADDR OF END REG
768	1233	1570	C=REGN	13	
769	1234	74	ROR	3	
770	1235	506	A=A+C	X	A.X=ABS ADDR OF END REG
771	1236	630	C=M		
772	1237	406	? A<C	X	END REG OVER LIMIT ?
773	1240	463	GONC	DATSER (1206)	YES
774	1241	606	A=A-B	X	B.X=ABS ADDR OF STARTING REG
775	1242	37	GOC	DATS40 (1245)	# OF REG < 0
776	1243	1506	? A#0	X	# OF REG = 0 ?
777	1244	27	GOC	DATS45 (1246)	NO
778	1245	6	A=0	X	
779	1246	546	A=A+1	X	
780	1247	1610	S0=	1	CHECK EOF
781	1250	256	AC EX	W	C.X= # OF REGS TO READ OR WRITE
782	1251	74	ROR	3	SAVE IN M.X= # OF REGS
783	1252	306	C=B	X	M.M= 1ST REG ADDR
784	1253	674	ROR	11	
785	1254	530	M=C		
786	1255	132	C=0	N	
787	1256	1	GOSUB	RG-BY#	# OF REGS TO # OF BYTES
787	1257	0			
788	1260	1170	C=REGN	9	GET # OF BYTE PASSED BOF
789	1261	516	A=A+C	W	
790	1262	212	B=A	WPT	B[4:0]= TOTAL # OF BYTES FROM BOF
791	1263	260	C=N		C[3:0]= # OF REGS OF THE FILE
792	1264	1	GOSUB	RG-BY#	
792	1265	0			
793	1266	1452	? A<B	WPT	WILL CROSS BOUNDARY THIS TIME?
794	1267	1640	RTN NC		NO
795	1270	1614	?S0=1		NEED TO CHECK EOF ?
796	1271	1	GOLC	CSEOF	YES, SAY "END OF FILE"
796	1272	3			
797	1273	260	C=N		READ WHAT EVER IN THE FILE
798	1274	406	A=C	X	A.X= FILE SIZE IN REGS
799	1275	630	C=M		
800	1276	246	AC EX	X	
801	1277	530	M=C		
802	1300	1740	RTN		

804	1301	DATALL	1	GOSUB	FNDEND	FIND ADDR OF LAST REG
804	1302		0			
805	1303		116	C=0		
806	1304		1160	DADD=C		
807	1305		1150	REGN=C	9	WRITE STARTING FROM BOF
808	1306		1570	C=REGN	13	
809	1307		74	ROR	3	C.X= REG0
810	1310		346	B=C	X	
810	1311		306			
811	1312		706	A=A-C	X	A.X = # OF REG TO WRITE
812	1313		1506	? A#0	X	SIZE=0 ?
813	1314		1347	GOC	DATS50 (1250)	NO
814	1315		1	GOLONG	DATSER	ERROR
814	1316		2			

*
*
* GVBREN - SAVE CURRENT BYTE POINTER AND READ LAST FILE ENTRY
* FROM DIRECTORY BUFFER

```

* INPUT : LAST ACCESSED FILE ENTRY PTR IS SAVED IN DIRECTORY BUFFER
*          AT BYTE #250. IF THIS # = 250, LAST OPERATION WAS NOT
*          A READ OR WRITE TO A DATA FILE
* OUTPUT : REG.9[4:2] = CURRENT RECORD #
*          REG.9[1:0] = CURRENT BYTE NUMBER
*          G = BYTE #251 IN DIR. BUFR.
*          IF G=0 LAST OPERATION ON A DATA FILE WAS READ
*          IF G=1                //                WRITE

```

```

*      826          ENTRY  SVBREN
*
830 1317 SVBREN      1 GOSUB  REQADR      READ CURRENT ADDR
830 1320              0
831 1321          256 AC EX  W          C[3:0] = CURRENT RECORD #
832 1322          1574 RCR      12
833 1323          312 C=B      WPT      C[1:0] = CURRENT BYTE #
834 1324          1150 REGN=C  9
835 1325          460 LDI
836 1326          372 CON      250
837 1327          1 GOSUB  SETBPC      SET BYTE PTR TO 250
837 1330              0
838 1331          1 GOSUB  RDLPBK      SEND CMD- READ LOOP BACK
838 1332              0
839 1333          1 GOSUB  NATNRD      SEND NAT
839 1334              0
840 1335          1 GOSUB  RDDFRM      READ LAST FILE ENTRY PTR
840 1336              0
841 1337          1200 HPIL=C  2          ECHO
842 1340          346 BC EX  X          SAVE ENTRY PTR IN B.X
843 1341          1 GOSUB  NRD        READ NEXT BYTE IN BUFFER TOO
843 1342              0
844 1343          1634 PT=      0
845 1344          130 G=C
846 1345          146 AB EX  X          SAVE THIS BYTE IN G
847 1346          460 LDI          GET LAST FILE ENTRY PTR
848 1347          372 CON      250
849 1350          1546 ? A#C  X          ENTRY PTR = 250 ?
850 1351          1 GOLNC  FLTYER      YES, NOT A VALID ENTRY PTR
850 1352              2
851 1353          1 GOSUB  SETBFL      PUT THE ENTRY PTR BACK
851 1354              0
852 1355          1 GOSUB  RDLPBK      SAY READ LOOP BACK
852 1356              0
853 1357          1 GOSUB  RENT10
853 1360              0
854 1361          1 GOLONG  RSTBP
854 1362              2

```

```

*
* SVENTR - SAVE FILE ENTRY PTR IN DIR BUFR BYTE #250 AND REMEMBER
*          LAST OPERATION TO A DATA FILE WAS A READ IN BYTE #251
* SVENTU - SAME AS SVENTR EXCEPT LAST OPERATION IS WRITE
*

```

```

860          ENTRY  SVENTR
861          ENTRY  SVENTW
862          ENTRY  SVMODE
863 1363 SVENTR 1604 S0=      0
864 1364          23 GOTO  SVENT  (1366)
865 1365 SVENTU 1610 S0=      1
866 1366 SVENT      1 GOSUB  REQADR      GET CURRENT TAPE ADDR
867 1367              0

```

```

867 1370      1410 S1=      1      NOT ONLY SAVE MODE
868 1371      146 AB EX    X      A.X = BYTE #
869 1372      460 LDI
870 1373      40 CON      32
871 1374      706 A=A-C    X      BACK UP 32 BYTES
872 1375      206 B=A      X
873 1376      460 LDI
874 1377      372 CON      250
875 1400 SVEN10 1 GOSUB    SETBPC    SET BYTE PTR TO 250
875 1401      0
876 1402      1 GOSUB    WRLPBK     SEND CMD- WRITE LOOP BACK
876 1403      0
877 1404      1414 ?S1=1
878 1405      43 GONC     SVEN20 (1411) YES
879 1406      306 C=B      X
880 1407      1 GOSUB    SDATA0     SENT B.P. OF CURR. FILE ENTRY
880 1410      0
881 1411 SVEN20 106 C=0      X
882 1412      1614 ?S0=1
883 1413      23 GONC     **+2      (1415)
884 1414      1046 C=C+1    X
885      LEGAL
886 1415      1 GOLONG    SDATA0
886 1416      2

```

```

*
888 1417 SVMODE 1404 S1=      0
889 1420      460 LDI
890 1421      373 CON      251
891 1422      1563 GOTO    SVEN10 (1400)

```

```

*****

```

```

*-INTCAL= CALCULATE INTEGER= INT [A*C + 0.5]

```

```

*
*-USES:      A,B,C,M,      PT,      S5,      2 SUB LEVELS
*-INPUTS:    A= (NNN-1)/(MAX-MIN)= FLOATING POINT [INTCAL ONLY]
*            C= (X-MIN) OR (-MIN) [INTCAL]
*            DEC MODE [INTCAL ONLY]
*-OUTPUTS:   C(X)= BINARY NUMBER
*            HEXMODE, DOESN'T CHANGE CHIP ENABLE
*

```

```

903      ENTRY    INTCAL
904 1423 INTCAL 1 GOSUB    MP2-10      ( ) (NNN-1)/(MAX-MIN)
904 1424      0
*CAN T OVERFLOW, AND UNDERFLOW RTNS C=0 WHICH IS OK, SO DON'T CHECK.
906      1 GOSUB    OVFL10      NORMALIZE UNDERFLOW TO 0
907 1425      0
907 1426      416 A=C      A=( ) (NNN-1)/(MAX-MIN)
908 1427      36 A=0      S      TAKE ABSOLUTE VALUE
909 1430      116 C=0
910 1431      1246 C=-C-1    X      EXP= -1
911 1432      520 LC      5      C= 0.5
912 1433      1 GOSUB    AD2-10     C= A + C
913 1434      0
913 1435
*CAN T OVERFLOW SINCE 0.5 ADDS NOTHING TO "9 E99"
915
916 1436      216 S5=      1      GET INTEGER PART
917 1437      1 GOSUB    INTERC     INT [C + 0.5]
917 1440      0
918 1441      1140 SETHEX

```

919		ENTRY	CONV3D	
920		ENTRY	CONV3C	
921	1442	CONV3C	406 A=C	X COPY EXPONENT TO A.X
922	1443		136 C=0	S TAKE ABSOLUTE VALUE
923	1444		106 C=0	X INITIALIZE ANSWER TO 0
924	1445		1372 ? C#0	M HANDLES ZERO UNNORML #S
925	1446		1640 RTN NC	
926	1447		1526 ? A#0	XS NEGATIVE EXPONENT?
927	1450		1540 RTN C	YES, ANSWER IS 000
928	1451		1574 RCR	12 MOVE 1ST DIGIT TO C.0
929	1452		646 A=A-1	X WAS EXPONENT 0?
930	1453		107 GOC	XGOTI (1463) YES
931	1454		1374 RCR	13 ROTATE NEXT DIG IN
932	1455		646 A=A-1	X WAS EXPONENT 1?
933	1456		57 GOC	XGOTI (1463) YES
934	1457		1374 RCR	13 MOVE 3RD DIGIT
935	1460		646 A=A-1	X WAS EXPONENT 2?
936	1461		1 GOLNC	ERRDE NO, NUMBER TOO LARGE
936	1462		2	
937	1463	XGOTI	1 GOLONG GOTINT	CONVERT ANSWER TO BINARY
937	1464		2	

*
*
* CONV3D - CONVERTS THE THREE DIGITS TO THE LEFT OF THE DECIMAL POINT
* IN THE X REGISTER TO A BINARY NUMBER AND LEAVES IT IN C.X
* CONV3C - SAME EXCEPT INPUT IS IN C.
* IGNORES THE SIGN OF X. IF ABS X IS GREATER THAN 999, GIVES "DATA
* ERROR". IF X CONTAINS AN ALPHA STRING, GIVE "ALPHA DATA" ERROR.
* IF A NON-NORMALIZED NUMBER WITH ZERO MANTISSA RETURNS ZERO.
*
* ASSUMES CHIP 0 ENABLED AND HEXMODE.
* USES A.X AND C, NO PT, NO STS, 1 ADDITIONAL SUB LEVEL (GOTINT)
* USUALLY EXITS VIA GOTINT IN THE MAINFRAME.
* RETURNS ANSWER IN C.X
*

952	1465	CONV3D	1 GOSUB	ACKX	GET X, ERROR IF ALPHA
952	1466		0		
953	1467		1533 GOTO	CONV3C (1442)	

954

*-PCHKKB= CHECK KEYBOARD (PRINTER FUNCTION)
*

*-RETURNS ONLY IF NEITHER THE "R/S" KEY NOR THE "ON" KEY IS DOWN
*

*-INPUTS: NONE

*-OUTPUTS: NONE

*-USES: A(X), C, NO PT, NO STS, NO ADDITIONAL SUB LEVELS
*

956		ENTRY	PCHKKB	
967		ENTRY	PCKBRT	
960	1470	PCHKKB	1714 CHK KB	KEY DOWN?
960	1471		1640 RTN NC	NO
970	1472		1040 C=KEYS	YES, GET KEYCODE
971	1473		74 RCR	3 KEYCODE TO C(X)
972	1474		126 C=0	XS
973	1475		406 A=C	X KEYCODE TO "A"
974	1476		460 LDI	
975	1477		30 CON	24 HEX 18= "OFF" KEY

976	1500	1546	? A#C	X	"OFF" KEY HIT?
977	1501	57	GOC	PCKB10 (1506)	NO
978	1502	1	GOSUB	IFC	
979	1503	0			
979	1504	1	GOLONG	OFF	
979	1505	2			
980	1506	PCKB10	460	LDI	
981	1507	207	CON	135	HEX 87= "R/S" KEY
982	1510	1546	? A#C	X	"R/S" KEY HIT?
983	1511	67	GOC	OUTPCK (1517)	NO
984	1512	PCKBRT	1304	S13=	0
985	1513	1	GOSUB	IFC	YES, CLEAR RUNNING FLAG
985	1514	0			
986	1515	1	GOLONG	NFRKB	RESET THE KEYBOARD
986	1516	2			
987	1517	OUTPCK	1710	RST KB	NO, TRY TO CLEAR THE KEY
988	1520	1740	RTN		

*

* CKANGL - CHECK IF CHARACTER WAS A ANGEL SIGN (ASCII 0D)
 * IF IT IS, REPLACE 0D BY 7C
 * REASON FOR THE CHANGE IS THAT, ASCII 0D IS A ANGEL SIGN IN HELIOS
 * BUT PIL PRINTER WILL USE 0D AS CARRIAGE RETURN SO WE HAVE TO MOVE
 * THE ANGEL SIGN FROM 13(0D) TO 124(7C).
 *

* USED B.X, N, PT, +0 SUB LEVEL
 *

999		ENTRY	CKANGL
1000		ENTRY	CKANGB
1001		ENTRY	CKANGN

*

1003	1521	CKANGL	206	B=A	X	SAVE A.X IN B.X
1004	1522	CKANGB	160	N=C		SAVE C IN N
1005	1523	CKANGN	406	A=C	X	PUT C.X TO A.X
1006	1524		1434	PT=	1	
1007	1525		460	LDI		
1008	1526		15	CON	13	
1009	1527		1552	? A#C	WPT	IS IT AN ANGEL ?
1010	1530		213	GONC	CKANG5 (1551)	YES
1011	1531		644	C=HPIL	6	
1011	1532		672			
1011	1533		603			
1012	1534		1166	C=C-1	XS	
1013	1535		1046	C=C+1	X	ARE WE TALKING TO T.V. ?
1014	1536		103	GONC	CKANG3 (1546)	NO
1015	1537		460	LDI		
1016	1540		176	CON	126	
1017	1541		1552	? A#C	WPT	IS THIS THE SIGMA SIGN ?
1018	1542		47	GOC	CKANG3 (1546)	NO
1019	1543		460	LDI		
1020	1544		34	CON	28	REPLACE BY ASCII CODE
1021	1545		63	GOTO	CKANG6 (1553)	
1022	1546	CKANG3	260	C=N		RESTORE C
1023	1547	CKANG4	146	AB EX	X	RESTORE A.X
1024	1550		1740	RTN		
1025	1551	CKANG5	460	LDI		
1026	1552		174	CON	124	REPLACE 13 BY 124
1027	1553	CKANG6	406	A=C	X	
1028	1554		260	C=N		RESTORE C
1029	1555		252	AC EX	WPT	

1030 1556 1713 GOTO CKANG4 (1547)

*
 * SEEK - SEEK TO A GIVEN RECORD
 * INPUT : A[3:0] = RECORD #
 * OUTPUT : CASSETTE AS A TALKER
 * SEEKN - SAME AS SEEK EXCEPT INPUT IS IN N[3:0]
 * SEEKRD - SEEK TO A GIVEN RECORD AND READ IT
 * INPUT : SAME AS SEEK
 * OUTPUT : CASSETTE AS A TALKER

*
 * USED A,C,S0-7 +1 SUB LEVEL
 *

1042	ENTRY	SEEK
1043	ENTRY	SEEKC
1044	ENTRY	SEEKN
1045	ENTRY	SEEKRC
1046	ENTRY	SEEKRD
1047	ENTRY	SEEKRN
1048	ENTRY	SEEK40
1049	ENTRY	SETBPT
1050	ENTRY	SETBFL
1051	ENTRY	SEEKR2
1052	ENTRY	SETBFO

1054	1557	SEEKR2	460	LDI		
1055	1560		2	CON	2	
1056	1561		103	GOTO	SEEKRC (1571)	
HF72	1057	1562	SEEKN	260	C=N	
1058	1563		474	ROR	8	
1059	1564	SEEKC	416	A=C		
1060	1565	SEEK	404	S8=	0	
1061	1566		53	GOTO	SEEK10 (1573)	
HF77	1062	1567	SEEKRN	260	C=N	
1063	1570		474	ROR	8	
1064	1571	SEEKRC	416	A=C		
1065	1572	SEEKRD	410	S8=	1	
1066	1573	SEEK10	1	GOSUB	LISTEN	SEND SEC.CMD - "SEEK"
1066	1574		0			
1067	1575		460	LDI		
1068	1576		244	CON	0244	
1069	1577		1	GOSUB	SCMD	
1069	1600		0			
1070	1601		256	AC EX		SEND RECORD # (TWO BYTES)
1071	1602		132	C=0	M	MAKE SURE REC.# < 512
1072	1603		1074	ROR	2	
1073	1604		416	A=C		
1074	1605		1	GOSUB	SDATA0	
1074	1606		0			
1075	1607		256	AC EX		
1076	1610		1574	ROR	12	
1077	1611		1	GOSUB	SDATA	SEND 2ND BYTE OF RECORD #
1077	1612		0			
1078	1613	SEEK20	1	GOSUB	CSSTAS	READ CASSETTE STATUS
1078	1614		0			
1079	1615		1	GOSUB	PLERCK	CHECK IF ANY ERROR
1079	1616		0			
1080	1617		214	785=1		STILL BUSY?
1081	1620		1737	GCC	SEEK20 (1613)	YES
1082	1621		114	784=1		ANY ERROR ?
1083	1622		1	GOLC	CSERR	

1083	1623		3			
1084	1624	SEEK30	414	?S8=1		NEED TO READ ?
1085	1625		137	GOC	SEEK40 (1640)	YES
1086	1626	SETBP0	6	A=0	X	SET BYTE POINTER TO 00
1087	1627	SETBPL	1	GOSUB	LISTEN	
1087	1630		0			
1088	1631	SETBPT	460	LDI		
1089	1632		243	CON	0243	SAD 03- SET BYTE POINTER
1090	1633		1	GOSUB	SCMD	
1090	1634		0			
1091	1635		246	AC EX	X	
1092	1636		1	GOLONG	SDATA0	
1092	1637		2			
1093	1640	SEEK40	460	LDI		
1094	1641		302	CON	0302	SAD 02 - READ (TALKER)
1095	1642		1	GOLONG	SCMDWT	
1095	1643		2			

*
*
* RSTBP - RESTORE BYTE POINTER
* INPUT : REG.9[2:0] = BYTE POINTER
* ASSUME: CHIP 0 ENABLE
* USED A,X,C +1 SUB LEVEL
*

1103		ENTRY	RSTBP
1104		ENTRY	SETBPC

1106	1644	RSTBP	1170	C=REGN	9
1107	1645	SETBPC	406	A=C	X
1108	1646		1613	GOTO	SETBPL (1627)

*
*
* CNTBYT - COMPUTE # OF BYTES BETWEEN TWO ADDR IN MM FORM
* INPUT : A[3:0] = STARTING ADDR
* B[3:0] = ENDING ADDR
* PT = 3
* OUTPUT : A[2:0] = # OF BYTES BETWEEN THE TWO ADDR
* USED A[3:0], B,X, C,X +0 SUB LEVEL
*

1110		ENTRY	CNTBYT
------	--	-------	--------

7FA11120	1647	CNTBYT	116	C=0	W
1121	1650		602	A=A-B	PT
1122	1651		43	GONC	CBYT10 (1655)
1123	1652		646	A=A-1	X
1124	1653		642	A=A-1	PT
1125	1654	CBYT05	642	A=A-1	PT
1126	1655	CBYT10	642	A=A-1	PT
1127	1656		37	GOC	CBYT20 (1661)
1128	1657		1046	C=C+1	X
1129				LEGAL	
1130	1660		1743	GOTO	CBYT05 (1654)
1131	1661	CBYT20	606	A=A-B	X
1132	1662		346	BC EX	X
1133	1663		246	C=A	X
1133	1664		406		
1134	1665		746	C=C+C	X
1135	1666		746	C=C+C	X
1136	1667		746	C=C+C	X
1137	1670		246	AC EX	X

```

1138 1671      706 A=A-C  X
1139 1672      446 A=A+B  X
1140 1673      1740 RTN

```

```

*
*
*
* SNBYTES - ROUTINE TO SEND A GIVEN NUMBER OF BYTES IN C-REG
* ASSUME - CASSETTE IS IN MIDDLE OF RECEIVING DATA
* INPUT - PT = NUMBER OF BYTES TO SEND -1
*         C(13:0)= LEFT JUSTIFIED BYTES TO SEND
* USED A, C, PT +1 SUB LEVEL

```

```

1150      ENTRY  SNBYTES
1152 1674 SNBYTES 1574 RCR      12
1153 1675      416 A=C      W
1154 1676      1 GOSUB  SDATA0
1154 1677      0
1155 1700      1624 ? PT= 0
1156 1701      1540 RTN C
1157 1702      1724 DEC PT
1158 1703      256 AC EX  W
1159 1704      1703 GOTO  SNBYTES (1674)

```

7F3C

```

*****
* PWRDN - SEND COMMAND "GROUP POWER DOWN"
*****

```

```

1166      ENTRY  PWRDN
1168 1705      216 CON      @216      N
1169 1706      4 CON      @04      D
1170 1707      22 CON      @22      R
1171 1710      27 CON      @27      W
1172 1711      20 CON      @20      P
1173 1712 PWRDN  1 GOSUB  PILEN      TURN THE CLOCK ON
1173 1713      0
1174 1714      1 GOSUB  PLERCK
1174 1715      0
1175 1716      344 HPL=CH 3
1176 1717      1 CH=      @000      SHUT OFF AUTO IDY
1177 1720      460 LDI
1178 1721      233 CON      @233      HEX 9B
1179 1722      1 GOLDNG SCMD
1179 1723      2

```

```

*****
*-BINBCD= CONVERT FROM BIN TO BCD
*
*-CONVERTS THE BINARY # IN A(X) TO BCD
*
*-USES:  A, B(S), C,      NO ST, ACTIVE PT,  1 ADDITIONAL SUB LEVEL
*              (GENMUM CALLS SUBROUTINE)
*-INPUTS: [BINBCD] A(X)= BINARY #,  A(S)= # OUTPUT DIGITS
*         [BINBCD] C(X)= BINARY #,  C(S)= # OUTPUT DIGITS
*         BOTH ENTRIES: HEX MODE
*
*-OUTPUTS: A(X)= DIGIT STRING (LEFT JUSTIFIED),  B(S)= # OUTPUT DIGITS
*         HEX MODE, LCD DISABLED, RAM DISABLED

```

```

*
1195          ENTRY BINBD0
1196          ENTRY BINBDC
1197          ENTRY BINBCD
7FD4 1198 1724 BINBD0 136 C=0 S
1199 1725 BINBDC 416 A=C
1200 1726 BINBCD 460 LDI
1201 1727          20 CON 020
1202 1730          1160 DADD=C
1203 1731          106 C=0 X
1204 1732          1760 PFAD=C
1205 1733          1 GO LONG GENNUM
1206 1734          2

```

OUTPUT 2,3, OR 4 DIGITS
INPUTS TO "A"

ADDR= CHIP 1 (NONEXISTANT)
UNSELECT RAM

UNSELECT PERIPHERALS
CALC BCD #, AND # DIGITS OUTPUT

```

*
1206
1209          FILLTO 01734
1210          EJECT

```

*

* DSWKUP - LOGIC FOR NORMAL WAKE UP FROM DEEP SLEEP

*

7FDD	1215	1735	DSWKUP	530	M=C	SAVE C REG	7FDD
	1216	1736		1	GOSUB WKUPLP	WAKE UP THE LOOP	
	1216	1737		0			
	1217	1740		1114	PS9=1	LOOP INTAKE ?	
	1218	1741		167	GOC KYCKX1 (1757)	NO	
	1219	1742		444	C=HFIL 4	GET SELECTED LOOP ADDR	
	1219	1743		472			
	1219	1744		403			
	1220	1745		404	SS= 0		
	1221	1746		1	GOSUB CHKLAD	VERIFY THE ADDR	
	1221	1747		0			
	1222	1750		1	GOSUB IFC	SEND IFC	
	1222	1751		0			
	1223	1752		1	GOSUB FNDPTR	SEE WHERE IS THE PRINTER	
	1223	1753		0			
	1224	1754		33	GOTO KYCKX1 (1757)		
	1225	1755		1	GOSUB SF5521	SET F 55&21, SET AUTOIDY PIT	
	1225	1756		0			
	1226	1757	KYCKX1	1670	C=REGN 14	RESTORE SS 0	
	1227	1760		1530	ST=C		
	1228	1761		630	C=M	RESTORE C	
	1229	1762		1	GOLONG RMCK10		
	1229	1763		2			
	1230				FILLTO @1763		
	1231	1764	PPAUSE	0	NOP	ENTRY FROM PAUSE LOOP	
	1232	1765	PRUN	0	NOP	RUNNING	
	1237	1766	WAKEP	1473	GOTO DSWKUP (1735)	WAKE UP FROM DEEP SLEEP W/O KEY	
	1234	1767	POWCFP	0	NOP		
	1235	1770	I/O5VP	0	NOP		
	1236	1771	DEEPS	1443	GOTO DSWKUP (1735)	WAKE-UP FROM DEEP SLEEP	
	1237	1772	COLDSP	1433	GOTO DSWKUP (1735)	COLD START ENTRY POINT	
	1238	1773	PRYID	10	CON @10	H	
	1239	1774		61	CON @61	I	
	1240	1775		23	CON @23	S	
	1241	1776		3	CON @03	C	
	1242	1777	CKSUMF	0	NOP	PRINTER CHECKSUM	

*

1244	UNLIST
1247	END

ERRORS : 0

SYMBOL TABLE

BINBCD	1726	-	
BINBD0	1724	-	
BINBDC	1725	-	
C8YT05	1654	-	1660
C8YT10	1655	-	1651
C8YT20	1661	-	1656
CHKC30	336	-	
CHKCST	335	-	
CHKPCT	414	-	
CKANG3	1546	-	1542 1536
CKANG4	1547	-	1556
CKANG5	1551	-	1530
CKANG6	1553	-	1545
CKANG8	1522	-	
CKANGL	1521	-	
CKANGH	1523	-	
CKSUMP	1777	-	
CHTBYT	1647	-	
COLDER	661	-	
COLDSP	1772	-	
CONV30	1442	-	1467
CONV30	1465	-	
COPYBF	306	-	363
CSERE0	400	-	
CSEREX	376	-	434 413
CSHCF0	364	-	340
CSR0Y	341	-	347
CSR0Y1	350	-	342
DATALL	1301	-	
DATS05	1212	-	1205
DATS10	1220	-	1222
DATS20	1223	-	1217
DATS40	1245	-	1242
DATS45	1246	-	1244
DATS50	1250	-	1314
DATSER	1206	-	1240
DAYSUB	1166	-	
DEEPSF	1771	-	
DFB0CK	1140	-	
DF0X10	1150	-	1146
DF0X20	1156	-	1153
DSPFER	405	-	
DSWKUP	1735	-	1772 1771 1766
FINDID	6	-	
FLPT10	420	-	
FNID10	13	-	52
FNID20	35	-	40
FNID30	41	-	36
FNID35	45	-	34
FNID40	57	-	51
FNID45	53	-	44
FNID60	62	-	
FNID65	67	-	56
FNID70	101	-	76
FNIDR1	60	-	102
IPDSVF	1779	-	

IFC	113	-	
INTCAL	1423	-	
INTDIR	361	-	
KYCKX1	1757	-	1754 1741
OUTPOK	1517	-	1511
PCHKKB	1470	-	
PCKB10	1506	-	1501
PCKBRT	1512	-	
POWCFP	1767	-	
PPAUSE	1764	-	
PRT1D	1773	-	
PRUN	1765	-	
PURGEF	135	-	
PURGEP	140	-	
PWEDH	1712	-	
RALL10	665	-	660
RDKY10	1025	-	1037
RDKY15	1032	-	1062
RDKY20	1035	-	1043
RDKY20	1040	-	1033
RDKY40	1044	-	1031
RDKY50	1062	-	1045
RDKYER	1076	-	1130
READA	606	-	
READK	1001	-	
REQA10	1111	-	1123
REQA15	1124	-	1121
RECADR	1100	-	
REMENT	145	-	
RSTBP	1644	-	
SEER	1565	-	
SEEK10	1573	-	1566
SEEK20	1613	-	1620
SEEK30	1624	-	
SEEK40	1640	-	1625
SEEKC	1564	-	
SEEKN	1562	-	
SEEKR2	1557	-	
SEEKRC	1571	-	1561
SEEKRD	1572	-	
SEEKRN	1567	-	
SETBP0	1626	-	
SETBPC	1645	-	
SETBPL	1627	-	1646
SETBPT	1631	-	
SNBYTS	1674	-	1704
STOP10	111	-	
SVSEEN	1317	-	
SVEN10	1400	-	1422
SVEN20	1411	-	1405
SVENT	1366	-	1364
SVENTR	1367	-	
SVENTW	1365	-	
SVMODE	1417	-	
VERF10	453	-	462
VERIFY	443	-	
NAKEP	1768	-	
URET10	160	-	151
URET15	171	-	
URET30	227	-	222

URET40	231	-	226	220
URET50	274	-	203	
URTA	467	-		
URTA10	500	-	506	
URTA15	543	-	540	
URTA20	571	-		
URTK	704	-		
URTK10	717	-	727	
URTK15	731	-	723	
URTK20	761	-	735	
XGOTI	1463	-	1456	1453

ENTRY TABLE

BINBCD	1726	-
BINB00	1724	-
BINBDC	1725	-
CHKCS0	336	-
CHKCST	335	-
CHKPCT	414	-
CKANG8	1522	-
CKANGL	1521	-
CKANGN	1523	-
CNTBYT	1647	-
COLDER	661	-
CONV30	1442	-
CONV3D	1465	-
COPYBF	306	-
CSERE0	400	-
CSEREX	376	-
CSNCFD	364	-
CSR0Y	341	-
DATALL	1301	-
DATSER	1206	-
DATSUB	1166	-
DFBDCX	1140	-
DSPERR	405	-
FINDID	6	-
FNID10	13	-
FNID60	62	-
IFC	113	-
INTCAL	1423	-
INTDIR	361	-
PCHKK3	1470	-
PCKBRT	1512	-
PURGEF	135	-
PURGER	140	-
PWRDN	1712	-
READA	606	-
READK	1001	-
RECADR	1100	-
REMENT	145	-
RSTBP	1644	-
SEEK	1565	-
SEEK40	1640	-
SEEK0	1564	-
SEEKN	1562	-
SEEKN2	1557	-
SEEKRC	1571	-
SEEKK3	1572	-
SEEKEN	1567	-
SETBPG	1626	-
SETBPC	1645	-
SETBPL	1627	-
SETBPT	1631	-
SHBYTS	1674	-
STOP10	111	-
SVDRN	1317	-
SVENR	1357	-
SVENTU	1365	-

SVMODE	1417	-
VERIFY	443	-
URET10	160	-
URET15	171	-
URTA	467	-
URTA20	571	-
WR1K	704	-

RDDFRM	1111	1335					
RDDFRM	1112	1336					
RDLPBK	1331	1355					
RDLPBK	1332	1356					
RDRG	632						
RDRG	633						
RDRG0	1071						
RDRG0	1072						
RDRG10	643						
RDRG10	644						
RDRGA	645						
RDRGA	646						
RENT10	1357						
RENT10	1360						
REGADR	145	306	1317	1366			
REGADR	146	307	1320	1367			
RG-BY#	532	740	1256	1264			
RG-BY#	533	741	1257	1265			
RMCK10	1762						
RMCK10	1762						
RSTBP	1361						
RSTBP	1362						
RSTKCA	1074						
RSTKCA	1075						
RWCHK	515	711					
RWCHK	516	712					
SCHDEV	6	111					
SCHDEV	7	112					
SCMD	314	1104	1577	1633	1722		
SCMD	315	1105	1600	1634	1723		
SCMD20	123						
SCMD20	124						
SCMDWT	1642						
SCMDWT	1643						
SDATA	300	1611					
SDATA	301	1612					
SDATA0	326	1407	1415	1605	1636	1676	
SDATA0	327	1410	1416	1606	1637	1677	
SEEX	160						
SEEX	161						
SEEX40	460						
SEEX40	461						
SEEXR2	361						
SEEXR2	362						
SEEXRD	154						
SEEXRD	155						
SEEXRN	445	623	1005				
SEEXRN	446	624	1006				
SEKSUF	547	746					
SEKSUF	550	747					
SETBPC	320	1327	1400				
SETBPC	321	1330	1401				
SETBPL	331	1353					
SETBPL	332	1354					
SETBPT	171						
SETBPT	172						
SF5521	1755						
SF5521	1756						
ENBYTS	177	232	241	250	254	264	272
ENBYTS	207	233	242	251	255	265	273

SHDRDN	577	
SHDRDN	600	
SHDRGA	557	572
SHDRCA	560	573
SHDRGC	575	
SHDRGC	576	
SRWRT	162	
SRWRT	163	
TALKER	13	1100
TALKER	14	1101
UNL	333	1206
UNL	334	1207
UNT	455	653
UNT	456	654
WAITS	302	
WAITS	303	
WKUP80	674	
WKUP80	675	
WKUFLP	1736	
WKUFLP	1737	
WPRON	470	
WPRON	471	
URLPBK	322	1402
URLPBK	323	1403
URTA20	757	
URTA20	760	

End of VASM assembly

EXTERNAL REFERENCES

ACKX	1465	
ACKX	1466	
AD2-10	1434	
AD2-10	1435	
AOUT1	11	
AOUT1	12	
ASN15	1057	
ASN15	1060	
BCDBIN	1175	1230
BCDBIN	1176	1231
BINED0	63	
BINED0	64	
CHKLAD	1746	
CHKLAD	1747	
CHKPCT	140	
CHKPCT	141	
CKMER	661	
CKMER	662	
CULDS1	667	
CULDS1	664	
CRIFL	545	
CRIFL	546	
CRIFL0	744	
CRIFL0	745	
CSEOF	1271	
CSEOF	1272	
CSEREX	772	
CSEREX	773	
CSERR	1622	
CSERR	1623	
CSSTAS	343	1613
CSSTAS	344	1614
DATSER	1315	
DATSER	1316	
DTFLOW	173	
DTFLOW	174	
ERR110	403	
ERR110	404	
ERRDE	1461	
ERRDE	1462	
ERRNE	1210	
ERRNE	1211	
ERRPR	503	
ERRPR	504	
FLINKA	500	
FLINKA	501	
FLSCH	511	705
FLSCH	512	706
FLSCH0	610	1003
FLSCH0	611	1004
FLSCH1	443	
FLSCH1	444	
FLSCH2	135	
FLSCH2	136	
FLTYER	1156	1351
FLTYER	1157	1352

FNDCAS	336			
FNDCAS	337			
FNDEND	517	612	1166	1301
FNDEND	526	613	1167	1302
FNDPTR	1752			
FNDPTR	1753			
GENHUM	1733			
GENHUM	1734			
GOTINT	1463			
GOTINT	1464			
GTFEND	476			
GTFEND	477			
IFC	1502	1513	1750	
IFC	1503	1514	1751	
INADRD	27			
INADRD	36			
INTERC	1437			
INTERC	1440			
LDSST0	667			
LDSST0	670			
LEFTJ	376			
LEFTJ	377			
LISTEN	310	1573	1627	
LISTEN	311	1574	1630	
MEMLFT	1011			
MEMLFT	1012			
MESSL	405			
MESSL	406			
MP2-10	1423			
MP2-10	1424			
MSG105	401			
MSG105	402			
NATNRD	1107	1333		
NATNRD	1110	1334		
NFRD	676			
NFRD	677			
NFRKB	1515			
NFRKB	1516			
NORMCK	1054			
NORMCK	1055			
NOROOM	621			
NOROOM	622			
NRD	1341			
NRD	1342			
NRDC	651			
NRDC	652			
NXTDEV	47			
NXTDEV	50			
OFF	1504			
OFF	1505			
OVFL10	1425			
OVFL10	1426			
PILEN	45	1712		
PILEN	46	1713		
PLERCK	15	345	1076	1615 1714
PLERCK	16	346	1077	1616 1715
PLERCK	364	420	761	
PLERCK	365	421	762	
RCL	60			
RCL	61			

XLIN	1464				
XLIN	1465				
XLINA	1567				
XLINA	1570				
/P870	1433				
/P870	1434				
DCARD	1752				
DCARD	1753				
DRG00	726				
DRG00	727				
ERRTN	1300				
ERRTN	1301				
MCK10	1762				
MCK10	1763				
STKB	1212				
STKB	1213				
STMS0	1700	1726			
STMS0	1701	1727			
TK200	633				
TK200	634				
TOPMT	773	1005	1202	1430	1755
TOPMT	774	1006	1203	1431	1756
RACK#	403	1240	1517		
RACK#	404	1241	1520		
RK#20	1645				
RK#20	1646				
STFLG	1220				
STFLG	1221				
JPCXSM	622	1310	1361		
JPCXSM	623	1311	1362		
JAITHD	1012	1332			
JAITHD	1013	1333			
JAT238	1014				
JAT238	1015				
JAT35	1344				
JAT35	1345				
JAT836	1347				
JAT836	1350				
JAT999	1150	1743			
JAT999	1151	1744			
JHATKY	1172				
JHATKY	1173				
JPCM	1750				
JPCM	1751				

End of VASM assembly